

NEWCASTLE UNIVERSITY

**Continuous Trust Management
Frameworks: Concept, Design, and
Characteristics**

by

Ahmad Alonaizi

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the

Faculty of Science, Agriculture and Engineering
Computing Science

August 2017

Abstract

A Trust Management Framework is a collection of technical components and governing rules and contracts to establish secure, confidential, and Trustworthy transactions among the Trust Stakeholders whether they are Users, Service Providers, or Legal Authorities. Despite the presence of many Trust Frameworks projects, they still fail at presenting a mature Framework that can be Trusted by all its Stakeholders. Particularly speaking, most of the current research focus on the Security aspects that may satisfy some Stakeholders but ignore other vital Trust Properties like Privacy, Legal Authority Enforcement, Practicality, and Customizability. This thesis is all about understanding and utilising the state of the art technologies of *Trust Management* to come up with a *Trust Management Framework* that could be Trusted by all its Stakeholders by providing a Continuous Data Control where the exchanged data would be handled in a Trustworthy manner before and after the data release from one party to another. For that we call it: Continuous Trust Management Framework.

In this thesis, we present a literature survey where we illustrate the general picture of the current research main categorise as well as the main Trust Stakeholders, Trust Challenges, and *Trust Requirements*. We picked few samples representing each of the main categorise in the literature of Trust Management Frameworks for detailed comparison to understand the strengths and weaknesses of those categorise. Showing that the current Trust Management Frameworks are focusing on fulfilling most of the Trust Attributes needed by the *Trust Stakeholders* except for the *Continuous Data Control Attribute*, we argued for the vitality of our proposed generic design of the Continuous Trust Management Framework.

To demonstrate our Design practicality, we present a prototype implementing its basic Stakeholders like the Users, Service Providers, Identity Provider, and Auditor on top of the OpenID Connect protocol. The sample use-case of our prototype is to protect the Users' email addresses. That is, Users would ask for their emails not to be

shared with third parties but some Providers would act maliciously and share these emails with third parties who would, in turn, send spam emails to the victim Users. While the prototype Auditor would be able to protect and track data before their release to the Service Providers, it would not be able to enforce the data access policy after release. We later generalise our sample use-case to cover various Mass Active Attacks on Users' Credentials like, for example, using stolen credit cards or illegally impersonating third-party identity.

To protect the Users' Credentials after release, we introduce a set of theories and building blocks to aid our *Continuous Trust Framework's* Auditor that would act as the Trust Enforcement point. These theories rely primarily on analysing the data logs recorded by our prototype prior to releasing the data. To test our theories, we present a Simulation Model of the Auditor to optimise its parameters. During some of our Simulation Stages, we assumed the availability of a *Data Governance Unit, DGU*, that would provide hardware roots of Trust. This DGU is to be installed in the Service Providers' server-side to govern how they handle the Users' data. The final simulation results include a set of different Defensive Strategies' Flavours that could be utilized by the Auditor depending on the environment where it operates.

This thesis concludes with the fact that utilising *Hard Trust Measures* such as DGU without effective Defensive Strategies may not provide the ultimate Trust solution. That is especially true at the bootstrapping phase where Service Providers would be reluctant to adopt a restrictive technology like our proposed DGU. Nevertheless, even in the absence of the DGU technology now, deploying the developed Defensive Strategies' Flavours that do not rely on DGU would still provide significant improvements in terms of enforcing Trust even after data release compared to the currently widely deployed Strategy: doing nothing!

For my beloved parents, passionate wife, and the three little treasures I am blessed with; my young daughters, I dedicate this thesis.

Acknowledgements

First and foremost, all the thanks and praises are due to Allah, the most gracious and the most merciful, for blessing me with a spirit that never stops seeking knowledge.

My warm acknowledgments then go to my supervisor, Prof. Aad van Moorsel, whose continuous guidance, and encouragements accompanied me throughout my PhD journey along with wise advice and kind attitude.

My acknowledgements also go to my dear friend and colleague, Dr. Suliman Al-suhibany for the valuable advice and encouragement he overwhelmed me with since my first day in the School. By letting me participate with him in some of his projects, I learned invaluable lessons and academic skills that I should never forget.

I extend my acknowledgements to everyone who spared time to listen to my ideas and gave valuable feedback. In particular, I mention my colleagues: Dr. Thomass Gross, Dr. Fransisco Rocha, and Dr. Maciej Machulak for their constructive discussion and evaluation. In fact, all my colleagues in Claremont Tower - Room 7.02 deserve the warmest greetings for making this place a social and brilliant place to pursue a PhD degree! Doing an extensive research project like a PhD requires a brilliant team of staff to take care of all the small details, so my appreciations go to all the staff in the CS department for their continuous support they always provide.

Spending over 4 years far away from home is insane without real friends who provide invaluable social support. My appreciations go to everyone who spared time to talk, socialize, support in the hard times, and have fun in the good times! Particularly speaking, my warmest greetings goes to Dr. Abdulaziz Alsharidah, Dr. Yousef Alqahim, Dr. Abdulaziz Alnahed, Hasan and Othman Alkandari for the valuable support they overwhelmed me with during my studies.

My acknowledgments and appreciations go then overseas to my home institute, the Public Authority for Applied Education and Training in Kuwait, PAAET, for giving me the opportunity to pursue the MSc and PhD degrees through their generous sponsorship.

I do not forget to warmly thank my parents who influenced me to seek knowledge and to devote myself to whatever I believe in. I extend my warm thanks to my wife who showed all the support and love that I desperately needed during my journey. Last but not least, my greetings are to my three little innocent girls who made my life exciting, happy, and full of action!

Contents

Abstract	iii
Dedictory	v
Acknowledgements	vii
Table of Contents	ix
List of Figures	xvii
List of Tables	xix
List of Equations	xxi
Glossary of Terms	xxiii
1 Introduction	1
1.1 Hypothesis	3
1.2 Methodology	4
1.3 Scope	5
1.4 Contributions	6
2 Trust and Trust Framework Requirements	9
2.1 Defining Trust and Trust Frameworks	10
2.2 Trust Stakeholders	11
2.3 Trust Life Cycle	12
2.4 Trust Challenges	13

2.5	Trust Requirements	15
2.6	Properties of the Trust Enforcement Requirement	16
2.6.1	Security	17
2.6.2	Privacy	20
2.6.3	Legal Authority Enforcement	23
2.7	Properties of the Trust Flexibility Requirement	27
2.7.1	Practicality	27
2.7.2	Customizability	29
2.8	Trust Measures Spectrum	30
3	The Need for Continuous Trust Management Frameworks	37
3.1	Current Trust Frameworks at a Glance	38
3.1.1	TCG	39
3.1.2	UCON	42
3.1.3	TAS3	46
3.1.4	PrimeLife	51
3.1.5	ABC4Trust	53
3.1.6	UMA	56
3.1.7	OAuth 2.0	58
3.1.8	OpenID Connect	59
3.1.9	Shibboleth	62
3.1.10	Kerberos	64
3.2	Evaluating the Current Trust Management Frameworks	65
3.2.1	The Trust Enforcement Properties and Attributes Supported by Current Trust Frameworks	65
3.2.2	The Trust Flexibility Properties and Attributes Supported by Current Trust Frameworks	73
3.2.3	Current Trust Frameworks on the Trust Measures Spectrum . .	76
3.2.4	Other Relevant Projects and Frameworks	80
3.3	Strengths of the Current Trust Management Frameworks	84
3.3.1	Strong emphasis on Security Attributes	85

3.3.2	Good realisation of the Privacy Attributes - except for the After Release Data Protection	85
3.3.3	Reasonable Practicality Attributes	86
3.4	Weaknesses of the Current Trust Management Frameworks	86
3.4.1	Lack of Continuous Data Control	87
3.4.2	Lack of solid implementation of the Legal Authorities Attributes	88
3.4.3	Lack of solid implementation of the Customizability Attributes .	89
3.5	Why a Continuous Trust Management Framework and What it Should Provide?	89
4	Designing the Continuous Trust Management Framework, Proto- type, and Problem Demonstration	91
4.1	Continuous Trust Management Framework Design	92
4.2	Continuous Trust Management Framework, the Simple Prototype . . .	96
4.2.1	Development Utilities	97
4.2.2	Problem Demonstrator, the Example Use-Case	97
4.2.3	Prototype Implementation Based on OpenID Connect	100
4.3	Generalising the Continuous Data Control Problem	105
4.4	DGU, the Hypothetical Data Governance Unit	106
4.4.1	DGU Sticky Policies	108
4.4.2	DGU Generic Design	109
4.5	Beyond the Continuous Trust Framework Prototype	111
5	Trust Auditor Conceptual Design and Algorithms	113
5.1	Basic Elements	114
5.2	TLRavg Ranking Approach	116
5.2.1	Gradient Probability Distributor	117
5.2.2	Local Trust Metric Record	118
5.2.3	Average of Local Trust Metric Record	119
5.3	TST Ranking Approach	120
5.3.1	Sole Testing Distributor	121
5.3.2	Sole Tester Ranking Agent	123

5.3.3	Sole Testing Metric Record	124
5.4	TGT Ranking Approach	126
5.4.1	Group Tester Ranking Agent	126
5.4.2	Simple Colluding Group Metric	128
5.4.3	Strong Colluding Group Metric	129
5.4.4	Weak Colluding Group Metric	131
5.4.5	Group Testing Metric Record	132
5.5	Global Trust Metric Record	137
5.6	Utility Units	140
5.6.1	SPs Popularity Record	140
5.6.2	Initial Interactions Record	141
5.6.3	Data Governance Unit	145
5.6.4	Trust Network Health Gauges	148
5.7	Basic Deployment Rules	153
5.8	Threat Model	156
5.8.1	Users not Reporting All Abuse Cases	157
5.8.2	Malicious Users Reports	157
5.8.3	Making Money by Fooling the Testing Agents	158
5.8.4	SPs Sharing Users' Credentials with MSPs Unintentionally	158
5.8.5	Reverse Engineering the Auditor Algorithms by an ME	159
5.8.6	Misleading the Auditor with Simple Data Abusing Algorithms	160
5.8.7	Colluding M(P)SPs	161
5.8.8	MPSP Committing Suicide Attack	165
5.8.9	MSPs Registration Bombarding	167
6	Trust Auditor Model, Simulation and Results	169
6.1	Simulation Scope	170
6.2	Development Environment	172
6.3	Simulation Model	173
6.4	Trust Model	177
6.5	Simulation Approach	183
6.5.1	Factors of Interest	184

6.5.2	DOE	189
6.5.3	Statistical Analysis	191
6.6	Experimental Stages	194
6.6.1	Stage 1: The no Auditor Case Followed by Initial Optimisation	195
6.6.2	Stage 2: 1st Optimisation Refinement of Attacks and Auditorial Settings	196
6.6.3	Stage 3: 2nd Optimisation Refinement of Attacks and Auditorial Settings	197
6.6.4	Stage 4: Comparing the Performance of Manual and Automatic Optimisation Settings	197
6.6.5	Stage 5: Introducing DGU with Offering Voluntarily DGU In- stallation Randomly along with a Refinement Iteration	201
6.6.6	Stage 6: Removing the Voluntarily DGU Installation Option along with a Refinement Iteration	204
6.6.7	Stage 7: Comparing the performance of vDGU and DGU	205
6.6.8	Stage 8: Evaluating the Different Defensive Strategies by ANOVA- Testing	206
6.7	Summary of the Optimized Defensive and Attacking Strategies	207
6.8	The Malicious Density Theory	210
6.9	Evaluating the Defensive Strategies' Flavours	217
6.10	Known Limitations and their Expected Effects	223
6.10.1	Unverified Trust Model	223
6.10.2	Limitations of the Utilized Simulation Cycle Concept	224
6.10.3	Biased Environmental Settings	225
6.10.4	Awkward Experimental Stages Design	226
6.10.5	Non-Linearity in the Measured Effects	227
6.10.6	Programming Errors	229
6.10.7	Human Mistakes in Updating the Experiments Settings	230
7	Discussions and Future Prototype Enhancements	231
7.1	Discussing the Proposed Continuous Trust Framework, Should it be Trusted?	232

7.1.1	Satisfying the Trust Stakeholders Needs	232
7.1.2	Enforcing Continuous Data Control	233
7.1.3	Enabling Real-Life Deployment Given the State of the Art Technologies	236
7.2	Discussing the Extents of the Posed Risks in the Threat Model	238
7.2.1	Effects of Users Not Reporting all Abuse Cases	238
7.2.2	Effects of Malicious Users	239
7.2.3	Effects of Sharing with Malicious SPs	239
7.2.4	Effects of Reverse Engineering the Detection Algorithms by a Malicious Entity	240
7.2.5	Effects of Misleading the Auditor with Simple Data Abusing Algorithms	242
7.2.6	Effects of Colluding MPSPs	244
7.2.7	Effects of Popular SPs Committing Suicide Attack Combined with MSPs Registration Bombarding	245
7.2.8	Effects of Introducing DGU	247
7.3	Enhancements to the Developed Prototype	248
7.3.1	Enhanced Defensive Strategies Release Order	248
7.3.2	Enhanced Malicious Density Theory	253
7.3.3	Enhanced DGU Deployment Rules	253
7.3.4	Enhanced Users Sticky Policies Options	254
7.3.5	Enhanced Ratings Publishing Mechanism	255
7.3.6	Enhanced Internal Ratings Records	255
7.3.7	Enhanced User Reporting Handling	256
7.3.8	Enhanced Testing Agents Performance	256
7.3.9	Enhanced Testing Agents Selection Algorithms	257

8 Conclusion and Future Directions 259

8.1	Future Directions	261
-----	-----------------------------	-----

A Screenshots of the Implemented Prototype 263

B	Initial List of the Simulation’s Process Factors of Interest	279
C	Detailed Milestones Data of the Simulation Stages	287
C.1	The Milestones of Stage 1: The no Auditor Case Followed by Initial Optimisation	287
C.2	The Milestones of Stage 2: 1st Optimisation Refinement of Attacks and Auditorial Settings	291
C.3	The Milestones of Stage 3: 2nd Optimisation Refinement of Attacks and Auditorial Settings	294
C.4	The Milestones of Stage 4: Comparing the Performance of Manual and Automatic Optimisation Settings	297
C.5	The Milestones of Stage 5: Introducing DGU with Offering Voluntarily DGU Installation Randomly along with a Refinement Iteration	306
C.6	The Milestones of Stage 6: Removing the Voluntarily DGU Installation Option along with a Refinement Iteration	310
C.7	The Milestones of Stage 7: Comparing the Performance of vDGU and DGU	315
C.8	The Milestones of Stage 8: Evaluating the Different Defensive Strategies by ANOVA-Testing	319
	Bibliography	341

List of Figures

2.1	Digital Trust Life Cycle	13
2.2	Trust Requirements	15
2.3	Trust Measures Spectrum	31
2.4	A New Venn Of Access Control For The API Economy [1]	35
3.1	The Current Picture of The Trust Frameworks/Protocols Literature	38
3.2	UCON Model Components [2]	43
3.3	A Components Overview for the TAS3 Architecture [3]	47
3.4	A High-Level Architecture for the PrimeLife Privacy Language [4]	52
3.5	Architecture for Attribute-based Credential Technologies [5]	55
3.6	High-Level Overview of the UMA Protocol [6]	57
3.7	A General Protocol Suite of the OpenID Connect [7]	61
3.8	An Abstract Authentication Flow of OpenID Connect Protocol [8]	61
4.1	Continuous Trust Framework Design	93
4.2	The MVC Generic Design Pattern	100
4.3	Authentication Flow using OpenID Connect	102
4.4	Malicious Interaction Sequence Diagram	104
4.5	DGU Generic Architecture	110
5.1	Auditor Theoretical Building Blocks	114
6.1	The Simulation Model High-Level Architecture	174
6.2	Visualising the Observed Malicious Density Values during the Simulation Process	213
6.3	Approximate Density Threshold Point and Regions Boundaries	215

List of Tables

3.1	Comparing the Trust Enforcement Attributes of Figure 2.2 Supported by Current Trust Frameworks	72
3.2	Comparing the Trust Flexibility Attributes of Figure 2.2 Supported by Current Trust Frameworks	75
3.3	Themes Comparison of the Evaluated Trust Frameworks	79
3.4	Other Industry/Standard Bodies Led Projects	81
3.5	Other Academic Projects	83
3.6	Other Deployed Authentication and Authorisation Solutions	84
3.7	Other Governmental Initiatives	84
6.1	Final List of Factors of Experiment Settings	184
6.2	Final List of Environmental Factors	184
6.3	Final List of Users' Controlled Factors	184
6.4	Final List of SPs' Controlled Factors	185
6.5	Final List of Auditor's and IDP's Controlled Factors	186
6.6	Final List of Attackers' Controlled Factors	186
6.7	Final List of Monitored Outputs	187
6.8	Environmental Factors Real Life Interpretation and Effects	199
6.9	Optimal Defensive Strategy (Manual Vs. Automatic) at Different Populations Scenarios	200
6.10	Classifications of the Defensive Strategies	207
6.11	The Main Defensive Settings against the main Counter-Attacking Strategies	209
6.12	The Observed Malicious Density Values during the Simulation Process	212
6.13	The Step Value Between Consecutive Density Points Based on the Density Measure	213

6.14 Evaluating the Defensive Strategies 221

B.1 Factors of Experiment Settings 279

B.2 Environmental Factors 280

B.3 Users' Controlled Factors 280

B.4 SPs' Controlled Factors 281

B.5 Auditor's and IDP's Controlled Factors 283

B.6 Attackers' Controlled Factors 284

B.7 Monitored Outputs 286

List of Equations

5.1	Gradient Probability Distributor, GPD	118
5.2	Local Trust Metric, TL	119
5.3	Average of Local Trust Metric, TL_{avg}	120
5.4	Sole Testing Distributor, STD	122
5.5	Sole Testing Trust Metric, TST	125
5.6	Group Trust Metric Rank for Gs, $TGT_{sg}(G)$	128
5.7	Group Trust Metric for SPs, $TGT_{sg}(SP)$	129
5.8	Strong Colluding Group Trust Rank, $TGT_{sc}(SP)$	130
5.9	Group Trust Rank, TGT(SP)	132
5.10	Global Trust Metric, TG	138
5.11	Recovery Period after Detecting an MPSP, $\tau_{MPSP-Halt}$	166
5.12	Recovery Period after Detecting MPSP assuming the MSPs Registration Bombarding Attack, $\tau_{MPSP-Halt}$	168
6.1	Trust increase in each cycle without an open Case	178
6.2	Trust decrease after receiving an Abuse Case	179
6.3	Trust decrease in each cycle with open Cases	179
6.4	Trust increase after resolving Abuse Case	180
6.5	Probability of generating a Credential at t	181
6.6	Probability of assigning a Strict Sharing Policy to a new Credential at t	182
6.7	Probability of assigning a Share with Any Policy to a new Credential at t	182
6.8	Basic Malicious Density Equation	211
6.9	Malicious Density Equation - PSPs	211
6.10	Malicious Density Equation - (P)SPs	211
6.11	Malicious Density Equation - PSPs and Users	211

7.1 Recovery Period after Detecting MPSP launching a combined Suicide and
MSPs registration Bombarding Attacks Considering the Soft Trust effects,
 $\tau_{MPSP-Halt}$ 245

Glossary of Terms

Term	Description
<i>Auditor Agent or Testing Agent</i>	An artificial User that is created by the Auditor to deal with suspicious (P)SPs to verify perceived doubts about them.
<i>Auditor</i>	An internal governing unit of the <i>Continuous Framework</i> to process Credentials Abuse reports, Rank SPs, and ban or punish suspected M(P)SPs.
<i>Case or C</i>	An Abuse Case containing a list of all the (P)SPs that have been dealt with since the last Case submitted by a User for the Auditor to detect the M(P)SP who is responsible for it.
<i>Colluding M(P)SPs</i>	A group of M(P)SPs collaborating to fool the Auditor's Defensive Strategy by sharing their transactions history to deduce whether a User they intend to Abuse is just an Auditor Agent or to make sure that the targeted User have dealt with a large enough set of M(P)SPs that makes it hard for an Auditor to detect the colluding group.

Counter-Attacking Strategy A counter-attacking pattern that could be adopted by the malicious Attackers to compromise the deployed Defensive Strategy of the *Continuous Framework's Auditor* in the quest to abuse Users' Credentials. The Main identified *Counter-Attacking Strategies* are: *Weak Colluding* and *Strong Colluding*.

Credentials or Personally Identifiable Information (PII) Small, but valuable, pieces of information that uniquely identify an entity like, for example, email addresses, physical addresses, phone numbers, credit cards numbers, passwords, and so on.

Defensive Strategy A defensive pattern that could be adopted by the *Continuous Framework's Auditor* to defend against malicious Attackers trying to abuse Users' Credentials. The main identified Defensive Strategies are: *Random*, *Conservative*, and *Aggressive*.

Defensive Strategy Flavour A customized combination of *Ranking Algorithms*, *Utilities*, and *Deployment Rules* settings that have the characteristics of its main Defensive Strategy, whether it is a *Random*, *Conservative*, or *Aggressive Defensive Strategy*.

Digital Trust A special case of general Trust where the Trust Stakeholders agree to engage in Digital Trust transactions with the "Intention to accept vulnerability based upon positive expectations of the intentions or behaviour of another [Stakeholder] [9]."

<i>DGU</i>	Data Governance Unit. A hardware utility to enforce Hard Trust in the Trust Network. If an SP <i>Installs DGU</i> , it should be impossible for it to handle an acquired Credential without compliance to its sticky policies. In addition, DGU would record all internal accesses to the Credential as well as all its approved sharing destinations.
<i>GPD</i>	Gradient Probability Distributor. A <i>Raw Ranking Algorithm</i> that Ranks SPs badly the closer the User have dealt with them just before getting an Abuse.
<i>GT</i>	Group Tester Agent. A <i>Raw Testing Agent</i> that interacts with a set of suspicious (P)SPs that are suspected to be colluding.
<i>Continuous Data Control</i>	Enforcing the Users' <i>Sticky Policies</i> before and after their release to requesting parties.
<i>Continuous Trust Management Framework</i>	Our proposed <i>Trust Management Framework</i> that should provide Continuous Data Control for its Stakeholders' data before and after their release to the other parties.
<i>Continuous Trust Management Framework Design</i>	A combination of the best practices we found in the current Trust Frameworks in addition to what we thought necessary and missing to present a generic design of a Continuous Trust Management Framework that fulfils the essential <i>Trust Requirements</i> with special attention to the <i>Continuous Data Control Privacy Attribute</i> .
<i>IDP</i>	Identity Provider.

<i>IIL</i>	Initial Interaction List. A list containing the interactions that took place by a given User's Credential prior to receiving the first Abuse on that Credential. This list would also include all the PSPs that this User have dealt with prior to that Abuse using other Credentials for detecting weak colluding attacks.
<i>IIR</i>	Initial Interactions Record. A Record containing all the IILs.
<i>Malicious Density Theory</i>	A theory describing the relation between the Density of the malicious nodes population in a Network and the effectiveness of the Auditor that tries to detect malicious activities.
<i>MGR</i>	Malicious Groups Record. A Record containing all the suspicious colluding groups. There are two variants of this Record to serve the Strong Colluding Algorithm MGR_{sc} and the Weak Colluding Algorithm MGR_{wc} .
<i>MPSP</i>	Malicious Popular Service Provider.
<i>MSP</i>	Malicious Service Provider.
<i>Personally Identifiable Information (PII)</i>	See <i>Credentials</i> .
<i>PIILs</i>	Popular Initial Interaction Lists. A Record containing the <i>IILs</i> that have appeared frequently among different Users triggering the possibility of colluding SPs.
<i>PSP</i>	Popular Service Provider.

<i>Ranking Approach</i>	A family of <i>Ranking Algorithms</i> that cooperate to generate a single Rank for each SP showing how Trustworthy it is based on the combined efforts of all the <i>Ranking Algorithms</i> of this family.
<i>Ranking Algorithm</i>	An Algorithm to process the aggregated data abuse records and, thus, assign each SP a Rank based on how Trustworthy it appears per the analysis.
<i>Selection Algorithm</i>	A descriptor for a family of Algorithms that analyze Users' logs to select suspicious (P)SPs to be processed by <i>Raw Agent Ranking Algorithms</i> .
<i>Simple Data Abusing Algorithms</i>	A descriptor for a family of simple Uncolluding Attacking Algorithms where the Attackers try to fool the Auditor by simple Delaying the Abuse, Repeating the Abuse, and/or Ignoring to Abuse certain Users' or some of their Credentials.
<i>SPR</i>	SPs Popularity Record. An Algorithm that sort all the SPs by their popularity based on the frequency of their appearing in the reported Cases.
<i>SP</i>	Service Provider.
<i>STD(SP_x)</i>	Sole Testing Distributor. A <i>Raw Ranking Algorithm</i> that gives all the Cases' SPs an equivalent rank. The less SPs appearing in the Case, the lower this unified Rank would be.
<i>Sticky Policy</i>	A policy that is attached to given Credential detailing how its owner wish it to be accessed, shared, used, and/or stored.

- ST* Sole Tester Agent. A Raw Testing Agent that interacts with only one suspicious (P)SP that is suspected to be Uncolluding with other M(P)SPs.
- Testing Agent* See *Auditor Agent*.
- TGR* Global Trust Metric Record. A Record containing all the assigned global Ranks.
- TG(SP_x)* Global Trust Metric. A Final Ranking Algorithm assigned to an individual *SP_x*, *TG(SP_x)*, by combining the value of the three Final Ranks: *TLR_{avg}(SP_x)*, *TST(SP_x)*, and *TGT(SP_x)* with customized weights given to each of them.
- TGT* Group Trust Metric. A *Final Ranking Algorithm* assigned to either a group *G*, *TGT(G)*, or an individual SP, *TGT(SP)*, based on the size of the colluding group. The smaller the group size, the smaller the assigned Rank. There are three *Intermediate Ranking Algorithms* generating three different variations of this Metric: Group Trust Metric *TGT_g*, Strong Colluding Trust Metric *TGT_{sc}*, and Weak Colluding Trust Metric *TGT_{uc}*. The Final *TGT* value is generated by combining the value of the three intermediate Ranks with customized weights given to each of them.

<i>TGTR</i>	Group Trust Metric Rank Record. A Record containing all the assigned group Ranks. There are two versions of this Record: $TGTR(G)$ for groups Ranks and $TGT(SP)$ for individual SPs Ranks. Also, there are three variations of this Record for the use of the corresponding <i>Intermediate Ranking Algorithms</i> : Group Trust Metric $TGTR_g$, Strong Colluding Trust Metric $TGTR_{sc}$, and Weak Colluding Trust Metric $TGTR_{wc}$.
$TL_{avg}(SP_x)$	Global Average Trust Metric. A <i>Final Ranking Algorithm</i> generating a global Rank for a given SP_x by averaging the aggregated TLR_U values for that SP_x .
TLR_{avg}	The Average TLR_U s Record. A Record containing all the TL_{avg} Ranking values for all SPs.
TLR_U	Local Trust Metric Record. A Record containing all the TL_{U_i} Ranking values.
$TL_{U_i}(SP_x)$	Local Trust Metric. An <i>Intermediate Ranking Algorithm</i> generating the average of the aggregated STD values for a given SP in the context of a single User.
<i>Trust Attributes</i>	A set of the main Trust Attributes of each of the main identified Trust Properties of each of the Trust Requirements that we identified in the literature.
<i>Trust Challenges</i>	A set of the main <i>Trust Challenges</i> that are facing the current Trust Frameworks based on our literature survey.

Trust Management Framework A collection of technical components and governing rules and contracts to establish secure, confidential, and Trustworthy transactions among the Trust Stakeholders whether they are Users, Service Providers, or governing Authorities.

Trust Measures Level A specific level, or category, in the Trust Measures Spectrum

Trust Measures Spectrum A spectrum of different categorise of tools and solutions that are categorized based on how strictly they rely on proofs to enforce Trust in the Network ranging from Soft Trust to Hard Trust Measures Levels

Trust Model A rough mathematical model we developed to mimic the Trust Framework's Users regarding to their perceived Trust and how this Trust would be affected after receiving a Data Abuse or after the Auditor detected the Abuser.

Trust Network Health Gauges A set of sensors to provide feedback reports about the health status of the Trust Network. Such sensors would aid the Auditor in figuring out whether to alter its currently deployed defences and how harmful the effects of the current attacks are.

Trust Properties A set of the main Trust Properties that make up each of the main Trust Requirements that we identified in the literature.

Trust Requirements	A set of the main Trust Requirements that should be fulfilled by any Trust Management Framework to satisfy its Stakeholders that we came up with by analysing the main Trust Challenges along with the best practices deployed by the current Trust Frameworks to tackle those Challenges.
Trust Stakeholders	The main Stakeholders of our Continuous Trust Management Framework are: End Users, Service Providers, Legal Authority, the Framework's Auditor (Governing the Network's transactions), and Identity Providers.
<i>TST</i>	Sole Tester Trust Metric. A <i>Final Ranking Algorithm</i> generating a global Sole Testing Rank by averaging the aggregated <i>STD</i> values for a given SP coming from both the <i>STD</i> Algorithm and ST Agents.
<i>TSTR</i>	Sole Testing Trust Record. A Record containing all the <i>TST</i> Ranks values.
<i>Uncolluding M(P)SPs</i>	A single M(P)SP that launches its attack without relying on collation feedback from other M(P)SPs.
<i>User or U</i>	Ordinary User of the Trust Network.

Chapter 1. Introduction

TRUST is the magical enabler for any two, or more, parties to engage in successful interactions, transactions, or delegations, as we explore in Section 2.1. For a complex environment like the Internet, which is composed of heterogeneous sets of sites and services consumed by a heterogeneous mix of humans and automated agents, Trust should lie at the heart of such an environment. The more Trust the Stakeholders have in the environment, the more interactions will take place. Without Trust, such an environment would be paralyzed.

That raises the need for a Trust Management Framework that provides and manages a set of protocols and software units to enable establishing a Network where the participants could engage in Trustworthy interactions. The currently available Trust Frameworks cannot satisfy all the needs of their Stakeholders, described in Section 2.2. Hence, these Trust Frameworks cannot be truly Trusted by them, see our survey of the current Trust Frameworks in Section 3.1.

Security solutions are plentiful but while it is true that Security is an important Property of Digital Trust, it is not the only one. Enhancing the Security of online transactions alone does not generate Trust and, hence, would pose limitations on the nature of the transactions that the Trust Stakeholders would be willing to engage-in, see Subsection 2.6.1. Therefore, the aim of this thesis is to define and evaluate the fundamental building blocks for a practical Continuous Trust Management Framework that satisfies all its Stakeholders. Particularly speaking, we are focusing in supporting the vitally lacked feature in the literature: the Continuous Data Control as we argue in Section 3.5.

The rest of this Chapter presents the main hypothesis, methodology, scope, and the main contributions.

In Chapter 2, we present a technical background on the subject to introduce the concept of Trust and Trust Stakeholders along with a detailed study of the current Trust Challenges, the corresponding Trust Requirements, and the Trust Measures Spectrum of the available tools and protocols for the current Trust Frameworks to deploy.

In Chapter 3, we present a thorough study of the literature where representative Trust Management Frameworks samples for each Trust Measures Level, see Section 2.8, is evaluated. That is followed by a thorough comparison of those sample Trust Frameworks in terms of how effectively they fulfil the Trust Requirements we list in Section 2.5. That is followed by a discussion of the strengths and weaknesses of the evaluated Trust Frameworks. From this discussion, we conclude with the need for Continuous Trust Management Frameworks along with a list of the minimal Trust Requirements they should provide to be qualified for the *Continuous* prefix.

In Chapter 4, we present the high-level design of our proposed Continuous Trust Management Framework along with a sample prototype to demonstrate the practicality of our proposal given the currently available technologies. We also present in that Chapter the design of a hypothetical Data Governance Unit, or simply DGU, that could solve a lot of the Trust issues currently facing the Trust Networks.

In Chapter 5, we present a set of basic formal building blocks and deployment rules to be used by an Auditor Unit to detect malicious activities and enforce Trust. These building blocks include our novel idea of artificial Testing Users, or Agents as we call them from now on, and how they could boost the performance of the proposed Auditor.

In Chapter 6, we present a Simulation Model representing the interactions that we implement in the prototype of Chapter 4. We describe in that Chapter the Simulation Process where we used the Simulation Model to study the behaviour and optimize the defensive parameters of our proposed Auditor when operating in large Networks with variable environmental settings. The results we present in that Chapter include

a set of Defensive Strategies Flavours that are suitable for deployment in different environmental conditions against different launched Counter-Attacking Strategies.

In Chapter 7, we argue that our proposed Continuous Trust Framework Design is capable of satisfying all its Stakeholders given our presented prototype and optimized Defensive Strategies Flavours and, hence, prove our ideas are practical and provide significant improvements to Users overall Trust. In that Chapter, we also evaluate the performance of our Defensive Strategies Flavours and their immunity to the risks presented in the Threat Model of Section 5.8 along with a list of possible enhancements to improve our current Defensive Strategies Flavours.

Finally, In Chapter 8, we conclude this thesis with the outcomes of this study and the possible future directions.

1.1 Hypothesis

The world would be ideal if we can Trust that every other party have good intentions and would behave as expected. That is not the case and, hence, we have legislation that governs legal contracts that can be used to judge who have breached the contract terms in case of any dispute we have in public courts. To enforce such laws and facilitate the work of our public courts, policing authorities are needed in real-life. In cyber life, we need another form of policing authority and juridical system to foster Trust with a high degree of certainty. At the same time, these cyber authorities should be ethical in the sense that they do not overuse their digital powers to Abuse Users' Data or falsely accuse a Network resident of breaching a contract term. In technical terms, the Trust Framework should be able to ethically protect two digital commodities: Users' Credentials (PII) and Users' Data, see Subsection 2.6.2. This protection should be provided before and after these digital commodities are released per the release contract. While many current Frameworks offer reasonable protection for these digital commodities before release, there is little done to offer after release protection, see Subsection 3.4.1.

The hypothesis of this thesis: it is possible, by aid of the novel Auditor Unit presented in Chapter 5, and by utilising the current available Trust Frameworks in the literature, to come up with a practical Continuous Trust Framework. This Continuous Framework would significantly improve the *Users' Trust* level by imposing Continuous Data Control on Users' Data and Credentials. That is, Data protection before and after these digital commodities are released to the requesting Service Providers. By doing so, our proposed Continuous Framework would provide an important Trust Attribute that the current Trust Frameworks fail to provide: *Data Control After Release*.

1.2 Methodology

As stated before, the aim of this thesis is to define and evaluate the basic theoretical building blocks for a practical Continuous Trust Management Framework that satisfies all its Stakeholders. For that, it is crucial to thoroughly understand the general problem of Trust and, then, combine both novel theories and practical prototypes to show the extent of the problem, what the proposed Continuous Trust Framework Design could achieve, and how future Trust technologies could positively affect our proposed Continuous Framework. Our methodology consisted of the following main milestones:

- **Surveying the Literature:** This milestone is concerned with studying the role of Trust in human lives in general as well as in computing to get a holistic understanding of the extent of the Trust issues and their possible impacts. That is vital to define the main qualities of Trust Framework's Role, Scope, Stakeholders, Challenges and Requirements. Moreover, a main goal of this milestone is to categorize the existing Trust Frameworks in the literature to understand their strengths and weaknesses as well as to utilize their best practices.
- **Designing Continuous Trust Management Framework:** This milestone is concerned with producing a generic Continuous Trust Framework Design based

on the best practices we observe in the literature survey milestone. This design should serve as a guideline for any future research in the area.

- **Proving the Practicality of the Continuous Framework Design:** This milestone is concerned with demonstrating the practicality of our proposed Continuous Framework by implementing a minimal prototype showing its main features and how its Stakeholders would interact. An essential goal of this milestone is to utilize the currently available Trust Frameworks in the literature to avoid reinventing the wheel. Particularly speaking, it should incorporate an Identity Service Provider, IDP, to manage the Users' identities and to keep access logs of them.
- **Defining Theories and Building Blocks to Construct the Auditor Unit:** This milestone is concerned with formally defining a set of novel theories and building blocks that would utilize the aggregated Users' logs by the IDP of the previous milestone. This Auditor Unit is to investigate any Data Abuse report to figure out who is guilty for the Abuse that happened after the Data released from its owner to the requesting Service Provider(s).
- **Simulating and Analysing the Performance of the Auditor Unit:** This milestone is concerned with analysing the performance and efficiency of the proposed Auditor. This milestone is about developing a Simulation Model to run and optimize the proposed Auditor in a large Trust Network with variable deployment settings.

1.3 Scope

It would have been great if this thesis could have covered every aspect of Digital Trust and fully implement the proposed Continuous Framework. Nevertheless, we are limited by the factors of time, resources, and experience. Hence, the scope of this thesis is as following:

- Thorough and deep, not exhaustive, literature surveying.

- Presenting a generic, not detailed, Continuous Framework Design based on the best practices we observe in the literature survey.
- Developing a simple, not complete or deployment-ready, prototype to demonstrate the practicality of our proposal.
- The presented Threat Model of Section 5.8 would describe many different attacking categories. However, the proposed Auditor Unit focuses only on tackling *Mass* and *Active Attacks* rather than *Targeted* and/or *Passive Attacks* due to their complexity, see Section 2.4.
- The Simulation Model implements a partial set of the proposed theories for the Auditor Unit to give preliminary results proving the hypothesis. Highly accurate results would require a real model that would be tested by real humans, which qualifies to be a separate research project by its own due to its complexity.
- Implementing specific core technologies that are present in the Continuous Trust Framework Design like advanced Sticky Policies or the DGU Unit is out of scope due to their complexity that may divert the development away from the original aim and hypothesis.
- While it is crucial for real-life deployments of the Trust Framework to have good QoS metrics such as scalability, resilience to Network conditions, and good responsibility speeds, neither our design nor our prototype implementation would focus on these aspects at this early stage of conceptualizing the Continuous Trust Management Framework.

1.4 Contributions

In this thesis, we introduce the following contributions:

- **Extensive Literature Survey:** an extensive survey covering the Trust related concepts, categorise, and some important example Frameworks as we show in Chapters 2 and 3.

- **Continuous Trust Management Framework Design:** a generic design that satisfy all its Stakeholders needs in a continuous manner, see Section 4.1.
- **Minimal Prototype of the Continuous Framework:** to prove the practicality of the proposed Continuous Trust Framework Design based on the current available technologies, see Section 4.2. The Code of this Prototype is published as an open-source project and can be found on: https://gitlab.com/Continuous_Trust_Prototype/Prototype.
- **Designing a Hypothetical Data Governance Unit, DGU:** this unit assumes the existence of mature TCG technologies, see Section 4.4. If the proposed design is implemented, it would significantly improve the Continuous Data Control Attribute as we show in our Simulation Process findings, see Subsection 7.1.2.
- **Novel Auditor Unit:** the theories, building blocks, and deployment rules for an Auditor Unit that can offer Continuous Data Control, see Chapter 5. Among these theories, we introduce the theories and governing rules to utilize artificial Testing Agents that would pose as real Users to prove whether a suspicious Service Provider is acting maliciously.
- **Developing Defensive Strategies Flavours for the Auditor:** this is a set of Defensive Strategies Flavours that resulted after many simulation refinement optimisation stages based on the three mainly identified Strategies: Random, Aggressive, and Conservative. These Flavours, are suitable for different environmental settings and against a variety of launched Counter-Attacking Strategies as we show in Sections 6.7 and 6.9. The Code of the Simulation Model of the Auditor and its interactions with the Trust Network Stakeholders, which we used to come up with our proposed Defensive Strategies is published as an open-source project and can be found on: https://gitlab.com/Continuous_Trust_Simulator/Simulator.

Chapter 2. Trust and Trust Framework Requirements

The aim of this research project and thesis is to define and evaluate the basic building blocks for a Continuous Trust Management Framework that could be Trusted by its Stakeholders by providing Continuous Trust where the exchanged Data would be handled in a Trustworthy manner throughout all the transactions phases. In this Chapter, we present the results of our survey and analysis of the rich literature of human Trust, in general, and Digital Trust, specifically, to answer the following fundamental questions:

- What is the definition of Digital Trust?
- Who are the digital Trust Stakeholders?
- How Digital Trust is established and how does its life cycle look?
- What are the current Trust Challenges?
- What are the Trust Stakeholders' main Trust Requirements?
- What are the main Trust Measures that are being utilized by the current Trust Frameworks?

Despite the wealth of novel ideas in the Digital Trust literature, there are still many dynamically evolving Challenges to be tackled. For that, we list in this Chapter the main Trust Challenges facing the current Trust Management Frameworks. These Trust Challenges are followed by their corresponding Trust Requirements where each Requirement has a set of Properties, which in turn have their own Attributes. Finally, we conclude this Chapter by illustrating the Trust Measures Spectrum categorizing the available tools and protocols at the disposal of the current Trust Managements

Frameworks into different Spectrum Levels ranging from Soft Trust Levels like *Cues and Clues* to Hard Trust Levels like *Hard Verification*.

2.1 Defining Trust and Trust Frameworks

Searching the word Trust in any search engine would reveal all sorts of theories, opinions, and practices that are related to almost any form of interactions whether it is among humans or machines. As Nissenbaum describes, **Trust**, in general, is a precious capital to any community, online or off, which flourish communication and transaction with a tolerance level to vulnerabilities that would be due to non-ill intentions [10]. A special forum discussing the view of Trust across different disciplines came up with the following definition: “Trust is a psychological state comprising the intention to accept vulnerability based upon positive expectations of the intentions or behaviour of another [9].” In his PhD thesis that tried to formalize a computational concept of Trust, Marsh described the Trust situation as “choosing to put ourselves in another’s hands, in that the behaviour of the other determines what we get out of a situation [11].”

Interactions involving computers or other technical means are no exceptions to the Trust rules. It is stated by Osterwalder that any technical development would rely, in its heart, on a form of Trust or delegation of Trust to other institutions [12]. Zhang et al. gave a more detailed description by thinking that **Digital Trust** is a subset mapping of the general Trust, where the specific subset and interpretation of Trust are dependent on the context [13].

For that, we define the **Trustworthy Transaction** to be any digital transaction that promise to fulfil a minimum set of **Trust Requirements**, see Section 2.5, that proves the good intentions of all the parties engaged in this transaction. Accordingly, we define the **Trust Management Framework** as a collection of technical components, governing rules and contracts to establish secure, confidential, and Trustworthy Transactions among the **Trust Stakeholders**, see Section 2.2, whether they are Users, Service Providers, or Legal Authorities.

So far, Trust has been described as a mental state enabling the Trust parties to engage in risky interactions where the expected loss, due to the failure of one party to fulfil its promises, usually outweigh the expected gain, if the involved parties acted as expected [11]. One may argue that Trust could be solely initiated by strong evidence and guaranties, which would remove the risk factor from the Trust definition. Nevertheless, achieving this high level of certainty is usually very expensive, if not impossible, in terms of the computational cost, limitations on the range of offered services, and the uncomfortable User experience as described by Nissenbaum [10]. A good Trust Management Framework should utilize a balanced set of Trust Measures from the **Trust Measures Spectrum**, See Section 2.8, that are neither too soft to be easily compromised nor too hard to be hostile for the Trust Stakeholders.

2.2 Trust Stakeholders

Trust Stakeholders are the main parties that are involved directly or indirectly in a Trustworthy Transaction, see Section 2.1. Depending on the context of the transaction, the Stakeholders could be Human Users, Software Agents, Legal Entities, State Authorities, or even a mix of all the previously mentioned groups. Nevertheless, most of the Stakeholders could be classified in one of three general groups: Service Providers, ordinary Users, and Legal Authorities. Each group of Stakeholders has its own needs and Requirements to engage in Trustful interactions.

For the Service Providers, it is important to authenticate the services' Users. Authentication could be as simple as an identity check but also could involve more complex analysis like credit history check. Moreover, some service providers wish to mine their Users' Data to generate personal profiles that could be used, for example, for targeted advertisements [14].

For the Users, the most precious commodities that they care about during any Trust relationship are their Credentials and Data. They wish to protect them in two different points of time: before and after releasing them to the second party. For that, it is important to verify the credibility of the online system to Trust using it. In addition,

some Users have concerns about the Security levels of the system while others would have concerns about the Privacy of their Data. As a result, many clients would have different Trust Requirements that they wish to negotiate with the Service Providers before engaging in any transaction.

For the Legal and Governmental Authorities, it is important to track cyber criminals, to monitor any transaction they suspect, and to enforce digital laws that are concerned, for example, with copyrights or fair trade competition. Such Requirements are very challenging to implement in practice due to the universal nature of the Internet where there is not a single authority or even country that governs the cyber web. The introduction of inter-linked systems, like cloud computing, just amplified the previous issues and added a new dimension of challenges [15]. Moreover, many Service Providers would refuse to collaborate with the Legal Authorities because they do not want to upset their customers who demand Privacy.

2.3 Trust Life Cycle

Per Nielsen, Trust is “hard to build and easy to lose [16].” As a result, it is important to understand the Trust building process and to picture how the Trust life cycle looks like. Zhang et al. described the Digital Trust building process as the problem of: initiating Trust, coming up with proper Trust metrics, aggregating feedback and propagating ratings, and setting a proper Trust Management Architecture [13]. Pearson and Benameur talked about the life cycle from a more abstract angle as they described three different phases in any Digital Trust relation: building Trust, maintaining stable Trust, and then losing Trust [15].

A general view of the Digital Trust Life Cycle, based on the current literature, is shown in Figure 2.1. The drawn cycle starts with building an appropriate Trust Framework to manage digital Trust in the desired context. Once the Trust Framework is ready, Trust Stakeholders could engage in Trust Requirements negotiation. Based on the agreed Requirements, proper Trust metrics are generated to provide solid ground to judge all parties’ conformance to the agreed contracts. After that, the Trust Framework

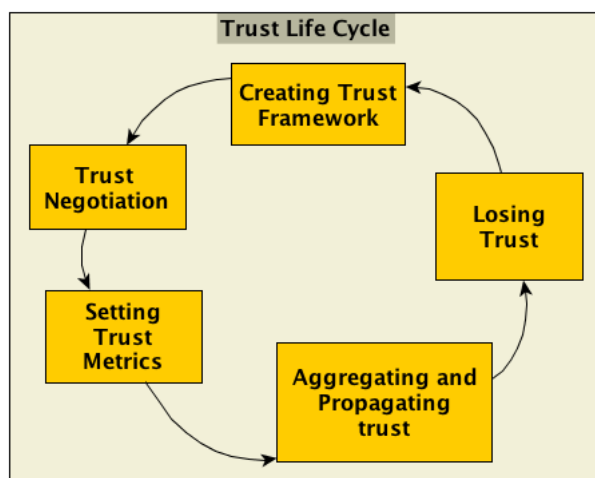


FIGURE 2.1: Digital Trust Life Cycle

should have the ability to aggregate useful feedback regarding the performance of all the engaged parties in the different Trusty interactions so it can prepare useful ratings and recommendations for any party wishing to engage in future transactions with any of the evaluated parties. If the Trust Management Framework failed to fulfil its promises or to keep up with its Users' expectations, it would quickly lose their Trust and, hence, a major update or a completely new Framework should be introduced to continue the Trust cycle.

2.4 Trust Challenges

As stated earlier in the introduction, Digital Trust does not equal digital Security. In fact, increasing the Security level does not necessarily imply Trust; it could be the opposite. Securing all transactions and communication channels may require revealing private or confidential Data that may increase the concerns of the participants about the usage of their personally identifiable Data, PII, which would lead to a distrust situation [12].

Per the 2010/2011 Computer Crime and Security Survey, CSI, cyber Security attacks could be categorized based on their coding themes [17]:

- **(Mass) Basic Attacks:** Phishing, ports scan, brute force attacks, old school viruses, and so on.
 - **Malware Attacks:** Extensions of the basic attacks utilising malware customisation and targeting toolkits.
- **(Targeted) Attacks 2.0:** Advanced Persistent Threat, APT, attacks where an organised committee of professional experts launch a digital attack targeting a specific entity with all the possible attacks at many different Security layers. Two recent examples are the Aurora and Stuxnet attacks.

Standard Security toolkits, like antiviruses and firewalls, could effectively minimise the risks of most of the Basic Attacks. However, that is not usually the case for the more advanced Malware Attacks that target specific entities with more complicated techniques, including social engineering. When it comes to the Targeted Attacks, the IT Security decision maker gets more confused due to the lack of visibility of the possible range of attacks that his enterprise might be exposed to or even face at any given time [17].

Another important classification of the possible Trust threats is based on the activity status, as explained by De Vivo in [18]:

- **Active Attacks:** The Attackers here would interfere in the performance of the compromised Network, trying to alter how it normally behaves. For example, erasing Data, modifying some entries, or causing denial of service.
- **Passive Attacks:** The Attackers here would not interfere in the performance of the compromised Network. Rather, they would simple sniff for the Data of their interest to utilize for their benefit, see Subsection 2.6.2. Unlike the Active Attacks, the Passive Attackers do not leave many foot-prints which makes their detection rather harder.

In many cases, a given attack could be described based on both above classifications. That is, an attack would normally be either a Mass or a Targeted Attack that is

indulging in Active or Passive attacking activities. When it comes to the advanced Targeted Attacks, it is expected that the Attacks would be a mixture of both Active and Passive Attacks. Such a mixture may also be utilized by some Mass Basic Attackers, specially the Malware extended version of it.

2.5 Trust Requirements

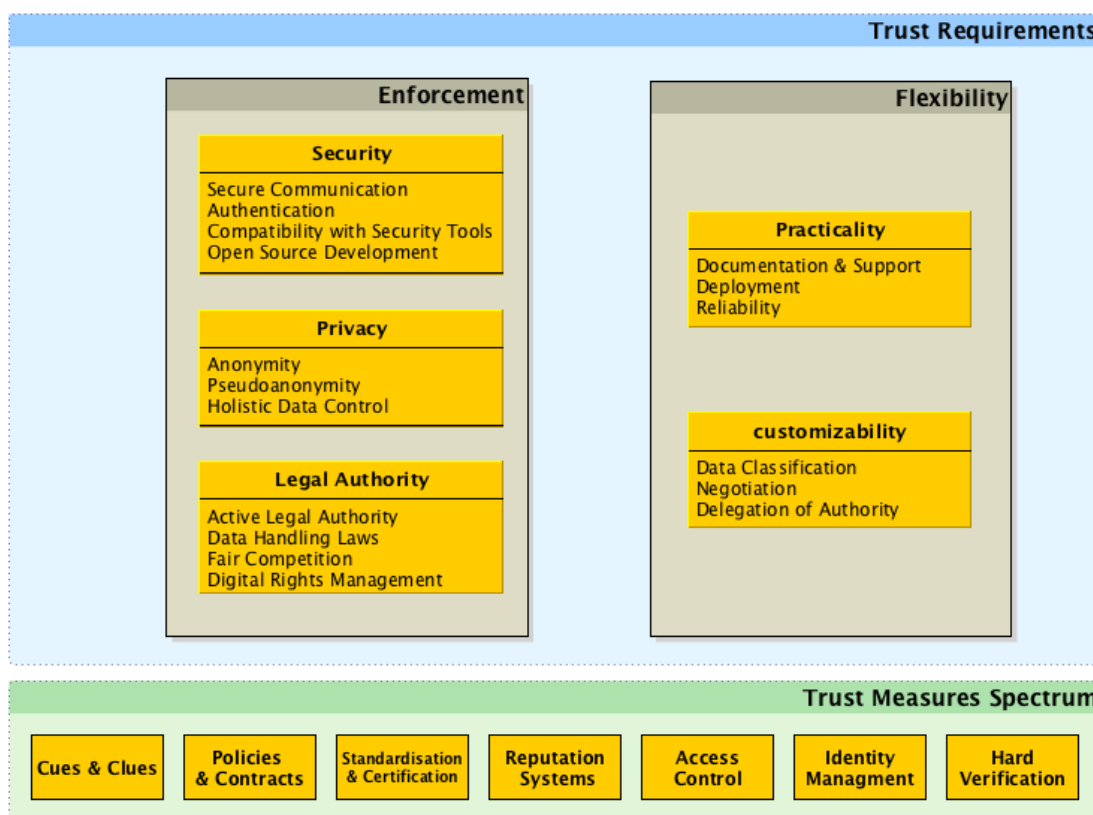


FIGURE 2.2: Trust Requirements

In this and the following Sections, we describe the main Trust Requirements that should be fulfilled by any Trust Management Framework to satisfy its Stakeholders. We came up with this list of Trust Requirements by analysing the main Trust Challenges we list in Section 2.4 along with the best practices deployed by the current Trust Frameworks to tackle these Challenges. In other words, whenever it becomes possible to tackle a Trust Challenge, by means of technical advancements, then tackling this Challenge would become a requirement rather than a mere wish by the Trust

Stakeholders. Not fulfilling such Requirements may lead the Trust Stakeholders to lose their Trust in the corresponding Trust Framework. That would be especially true in case of a breach of the Trust contract between two Trust Stakeholders since such incidents would make it clear that one of the Stakeholders is not doing its best to fulfil the Requirements of the others, see Section 2.3 for more details about the Trust Life Cycle.

Like in real-life transactions, any two or more parties would have two main Requirements to establish a Trustworthy contract:

- **Enforcement:** The ability to enforce the terms of the agreed upon contract by guarding the Trust transaction against malicious activities from outsider malicious entities as well as malicious activities conducted by some of the internal Trust Stakeholders against the agreed upon contract. Moreover, in case one party fails to deliver the quality of service it promised to in the contract, other parties should be able to resort to Legal Authorities for compensation.
- **Flexibility:** The ability to negotiate a tailor-made contract addressing the specific needs for the involved Stakeholders.

In Figure 2.2, we illustrate the two main Trust Requirements: Enforcement and Flexibility along with their main Trust Properties and those Properties' Attributes as well as the Trust Spectrum of the variety of Measures levels that could be deployed by the Trust Frameworks for the sake of fulfilling the Trust Requirements. More details about the Trust Spectrum can be found in Section 2.8. In the following Subsections, we describe the main Trust Requirements of Figure 2.2 in more details.

2.6 Properties of the Trust Enforcement Requirement

The first keyword that would come up to many readers' mind when mentioning the Trust Enforcement Requirement would be Security. It is true that Security is a crucial Property of the Trust Enforcement Requirement but this Requirement is more general

than just providing Security for the Trust Stakeholders. In fact, Security is like a safe guard for the communication and transactions among the Trust Stakeholders against outsider malicious entities. Nevertheless, Security does not necessarily guard against malicious entities that are part of the concerned Trust Stakeholders. That is, if one of the Trust Stakeholders decides to act maliciously, Security alone may not prevent him from doing so since he has unrestricted access to the exchanged Data among the Trust Stakeholders. To protect the exchanged Data from unauthorised malicious activities conducted by a malicious Trust Stakeholder, the Privacy Property is needed. In case the deployed Security and Privacy measures fail at preventing malicious activities whether these activities are from internal or external attackers, the Trust Stakeholders would need to resort to a Legal Authority that would have the power of law to gather all sorts of needed evidence, whether digital or physical. The gathered evidence would be used to investigate any dispute among the Trust Stakeholders to find out who is at fault and, hence, should compensate the other parties. For that, we discuss in the following Subsections these Trust Enforcement Requirement Properties, as they appear in Figure 2.2:

- Security
- Privacy
- Legal Authority Enforcement

2.6.1 Security

While Security is not the only Property of Trust Enforcement Requirement, it is definitely the most important one. Security is like the outer gate for the Trust Framework. If malware and Trojans exploit the Security gate, all the internal Trust Enforcement mechanisms could fail dramatically. Apart from the standard Security threats that could be tackled with general toolkits like firewalls and antiviruses, there are many device specific or even situation specific Security dimensions that should be tackled carefully. For example, the introduction of ordinary web browsers to the smart phones

introduced a new set of unanticipated threats. That is because the original design of the web browsers was to show web pages, and was not to directly operate physical resources like sending SMS or making a phone call [19]. Another example can be found in a non-software dimension. While software Security toolkits are the main components of digital Security, it is worthwhile to consider the physical Security of the enterprise buildings. Davis showed that most of the enterprises today rely on third-party companies to guard their physical buildings where the Data warehouses are located. Such third parties do not usually deploy advanced guarding mechanisms such as biometric logging and pc-proximity detectors, which would generate a new layer of possible physical attacks [20]. A tricky part of Security is reliability. If a software cannot scale well, it may crash at the first DoS attack which may cause irreversible losses to Users business or even Data. In addition to scalability, a software should be well tested and sanctioned, as part of the reliability requirement, to prevent leaving unintended Security holes for the malicious hackers. Below we list some of the primary Security Attributes, as they appear in Figure 2.2, that should be implemented by any Trust Framework that is serious about enforcing Trust, based on the best practices that are already present by well-established solutions in the literature and in the market as well.

- **Secure Communication:** This is a basic, yet crucial, Security Attribute to eliminate the threat of the man in the middle attack, MiM, where an outsider malicious entity may try to intercept the communication among the Trust Stakeholders to compromise the Trust Framework Security measures. This basic Attribute has been identified early on the literature with the well-known Kerberos protocol that establishes secure communication channels appearing in the literature since the late 1980s, see Subsection 3.1.10 for more details about Kerberos.
- **Authentication:** This is another basic, yet crucial, Security Attribute that requires any party trying to access a sensitive set of Data or engage in a transaction to be authenticated, i.e. to possess Credentials that prove she is allowed to do the action she is trying to perform. This is important to guard against

malicious outsider attackers trying to gain unauthorised access to the exchanged Data among the Trust Stakeholders. It is similarly important to prevent unauthorised employees working for an entity that is part of the Trust Stakeholders from gaining unauthorised access to the Data as agreed upon in the Trust contract. Moreover, this Attribute could help in logging who accessed what and when for the purpose of presenting evidence for the Legal Authority that may investigate any disputes among the Trust Stakeholders, our proposed Auditor automate this task as presented in Chapters 4 and 5. There are various ways to accomplish authentication in the literature like biometric measures, two steps verification, or simply possessing valid usernames and passwords, see Subsection 3.2.3 that shows a variety of access control techniques deployed by some of the Trust Framework that appears in the literature.

- **Compatibility with Security Tools:** This Attribute is about the Trust Framework ability to integrate with already existing legacy Security and Trust solutions. While a Trust Framework may be implemented completely from scratch to address all the Security and Trust Attributes, it would be prone to errors, due to the enormous size of the project, that would be easily mitigated by adopting the long established and polished solutions in the literature that are also maintained by dedicated teams of experts. In addition, entities willing to adopt such a new Trust Framework may find it challenging to abandon the current solutions they deploy in their servers for a fairly long time which would, in turn, cause a problematic bootstrapping phase for the new Trust Framework.
- **Open Source Development:** While the previous Security Attributes would not normally generate controversial debates among Security experts, this Attribute does. The advocates of open source development argue that it boosts the Security level of the produced software by opening it up for more experts to review it other than its authors, just like how peer-reviewed academic papers work [21]. Nevertheless, the opponents reply by questioning whether there are actually real experts taking the time to voluntarily review the open source software for Security vulnerabilities [22]. An empirical study by Guido suggests

that there is no significant advantage to either closed or open source software development approaches in terms of Security at the time of conducting their study in 2011 [23]. In other words, open source software, in theory, would boost the Security of the final product by letting external experts audit the produced code. In practice, there must be good motives for such experts to step in and do the auditing. Otherwise, the only experts who would review the code would be the malicious attackers trying to attack the system. Moreover, if there is any concern about malicious insider attackers, then closed source development should not be an option since the insider attackers could hide their malicious actions behind closed source code. For that, a Continuous Trust Management Framework that tries to ensure all parties about its fairness and neutrality toward the needs of all of them should be developed as an open source project so that all Trust Stakeholders who may have conflicting interests could check for themselves whether the project is fulfilling its promises or not.

2.6.2 Privacy

Data protection is a must for any Trust Framework to be truly Trusted by all of its Stakeholders. Data breachers usually look for financial sensitive Data for identity fraud or digital robbery crimes. This could be observed from the 2010 Verizon Security report which states that 33% of Data breaches in 2010 affected the financial sectors while 23% of breaches in that year affected the hospitality sector [24]. For that, providing proper Privacy measures for the exchanged Data among the Trust Stakeholders is a vital Property for all of them. Below is a list of the main Privacy Attributes, as they appear in Figure 2.2, that should be provided by a Continuous Trust Management Framework whenever they are needed:

- **Anonymity:** This Attribute is about concealing the identity of the owner of the Data set that a certain party have collected through Trusted communications and transactions. In some applications, this Attribute is a must in all the Trust transaction life phases. In other words, Users should be able to engage in a

Trust transaction without even revealing their identity, or any side information leading to conclude their identity. Such applications include voting, whistle blowing, seeking medical advice by persons who are too embarrassed to discuss their issues in public, and the like of these scenarios. In other applications, it is acceptable that the Service Provider identify the owner of the collected Data set but whenever the provider decides to share its collected Data sets with a third-party, with prior consent with the Data owners, then the provider should make every effort to conceal the identities of the Data owners. Such applications include, for example, commercial online stores requiring the detailed information of the customers to fulfil their orders. These stores may decide to share their log of transactions with a marketing firm to perform Data mining to explore new purchases trends or demographics of their customers. In such cases, the online stores should conceal the identities of its customers before sharing with the third-party firm. When anonymising the Data sets, it is not always sufficient to obscure the name of the customer. If it is, somehow, known that only one customer is living in a given area, then that customer could be implied in the transferred Data sets even if his name was obscured. For that, the online store could agree with its customers to provide a certain K-anonymity factor where at least K customers could be implied given the obscured Data sets where the larger the K value is, the more anonymised the customer would be [25]. While most traditional P2P service providers, for example, sacrifice Users' need for anonymity for the sake of improved Security [26], many newer Frameworks are focusing more on this important Privacy Attribute, see Section 3.1 for more details.

- **Pseudoanonymity:** This Attribute is about enabling the Trust Framework Users to have multiple identities when interacting within the Framework's environment. Just like in real-life, many of us would have a formal persona at the workplace that seems completely unrelated to our more relaxed and fun persona we have when we interact with friends and families. Likewise, we would have many different identities, or hats, depending on the context where we interact. In many cases, mixing up all the Attributes of our different identities to form a

new unified identity is not desirable at all. Most of us would not be happy if a potential employer starts inspecting our private Facebook or Twitter accounts where we may discuss controversial issues or simply exchange jokes that may be considered inappropriate. For that, many people tend to create pseudo-names whenever they need to get a new identity that is unrelated to their professional one and, for that, any Trust Framework that manages its Stakeholders identities should be able to offer pseudo-anonymity for them. This need has been already recognised by some of the current Trust Framework that we evaluate in this thesis such as Tas3, PrimeLife, and UMA in Subsections 3.1.3, 3.1.4, and 3.1.6 respectively.

- **Continuous Data Control:** While improved Security could guard the Trust Stakeholders' exchanged Data against outsiders' attacks, ordinary Users would still have valid concerns about whether their collected Data are handled properly by the Service Provider they Trusted to engage with. Moreover, some Users may even require at some instances to get their records completely removed from the Service Providers servers, a digital right that seems simple but, yet, many Service Providers fail to achieve specially in the cloud environment where there are multiple backup records for the same Data set [15]. As Joshi and Kuo describe, Service Providers face a trade-off between protecting Users' Data and selling these Data to advertisers to generate revenue [14]. Hence, ordinary Users need the ability to control what Data the Service Providers would collect from them and how those providers would handle their Data after they are acquired. We refer to this Privacy Attribute as Continuous Data Control. This control could be enforced by tailor-made contract negotiations, as we describe in Subsection 2.7.2, to control the flow of Users' Data before they are acquired by the Service Providers. Once Users' Data are acquired by the Service Providers, some form of technically sound assurance about the behaviour of the Service Provider is required. Such assurance could be generated by Hard Trust Level Measures that are based on evidence [27] or perhaps by softer Trust Measures Level that are based on Users' ratings and views of the Service Providers [9], see Section 2.8 for more details about the Trust Measures Levels. Despite the importance of

this genuine need, there is little done to address it in the literature and, hence, the need for our proposal for a Continuous Trust Management Framework, see Subsection 3.4.1 for more details.

2.6.3 Legal Authority Enforcement

This is another crucial Trust feature that is rapidly getting more mature and effective. It is true that less than 30% of the cyber-attacks incidents faced by the participants in the 2010 CSI survey were reported to the Legal Authorities because of the victims' believe that Legal Authorities can do little to improve the situation [17]. But it is also true that the sentencing of Albert Gonzalez for stealing about 170 million credit cards, for example, frightened his fellow hackers and raised the confidence in the power of the Legal Authorities [17]. Digital laws are necessary to protect the ordinary Users' interests in complex situations where, for example, Data control is lost in the cloud [15] or where non-competitive marketing schemes could be enforced by utilising technologies like the TPM chip [28, 29]. Below is a list of the main Legal Authority Attributes, as they appear in Figure 2.2, that should be provided by a Continuous Trust Framework whenever they are needed:

- **Active Legal Authority:** Interestingly, what makes the Legal Authority Enforcement Property of Trust Enforcement unique is the fact that it requires an Active Legal Authority that takes efforts and implements measures in addition to the Attributes provided by the Trust Framework. In other words, no matter how good the Trust Framework is designed and implemented, the Legal Authority Enforcement Property would not be completely satisfactory to all the Trust Stakeholders without the Legal Authorities being active by taking extra efforts and implementing certain measures to make effective Legal Enforcement. For example, legal acts and guidelines could aid the Service Provider in designing a better and more Trusted Framework as most participants in the 2010 CSI survey thought that compliance with regulatory laws helped them improve their Security [17]. The Security analysis of e-government portals in the US states

suggests a similar result [30]. Nevertheless, among the main challenges facing legislators today are keeping up-to-date with the rapidly evolving technologies such as TPM and cloud computing. In addition, legislators should try to make the lengthy laws more accessible and easy to comprehend by service providers who are currently confused about whether a particular law does apply to them or not [17].

- **Data Handling Laws:** This Attribute is about designing the Trust Framework so that it respects all the legal acts governing how it should handle any Data it acquires. That is a complex Attribute with different sets of laws that apply at different region where the Trust Framework operates. One of the key legalisation in this regard is the EU Directive for Data Protection which, since its application two decades ago, has acted as a role model for other legalisations around the globe. In addition, it forced indirect legal changes to comply with it since it restricts processing and retaining Data of EU residents to regions where their Privacy is protected per its terms [31]. The developed interest in protecting Users' Privacy online has led Cavoukian, the Privacy Commissioner in Ontario Canada, to come up with the 7 laws of identity which are nicely translated into software design requirements by the TAS3 developers [32]. Another good analysis of the legal and Trust Challenges facing the Cloud Frameworks today, which could be viewed as specialised forms of Trust Frameworks, is presented by Pearson and Benameur [15]. Of course the developers of any Trust Framework should carefully put in mind any specific laws that applies in the regions where they intend to operate their Framework. Nevertheless, some common rules based on the laws and studies we have listed here are likely to be common to many regions around the globe, due to globalisation effects where some countries would alter their own legalisation to facilitate trade and communication with other regions. The generic rules are as following:

- User's Consent prior to processing his Data
- No unnecessary Data processing, in complying with existing contracts or Legal Authorities request, should be allowed

- The Data processor, destination, purpose of access, and alteration should be logged and be available to its owner
- The Data collector should obtain the minimal amount of personal Data that is necessary to facilitate the consented Data operation
- The Data owner has the right to alter her shared personal Data or even to erase all the copies of such Data (the right to be forgotten)

An important special case where proper Data handling mechanism should be in place is where the Legal Authorities need to access protected Data by the Trust Framework to investigate a dispute or a crime. This operation is a subject for controversial debate between the proponents who argue that such access is vital to resolve open crimes and opponents who fear that possibly corrupted Legal Authorities may abuse their power to access unnecessary Data for any illegal purpose such as blackmailing certain people or to track individuals who may have ideologies or thoughts against the mainstream norms that are imposed by a totalitarian regime. The case of the San Bernardino terrorist's encrypted iPhone is a recent example showing how deep the conflict is between the Legal Authorities, the FBI in this case, who wish to access the protected Data and the Trust Framework admin, Apple in this case, who does not want to get distrusted by its Users if it corroborated with the Legal Authorities. Similar cases are expected in the future and it is likely that the legislators would demand Service Providers, such as Apple, to cooperate in enabling access to protected Data if demanded by the Legal Authorities [33]. For that, proper implementation of the Data Handling Laws would make the Trust Framework flexible enough to allow efficient access to the protected Data without revealing more than the required Data by law. In addition, all the Data access operations would be logged for the reference of the Data owners. That is necessary so that Users do not fear that the Service Providers are corroborating with possibly corrupted Authorities to undermine their freedom of speech or to exercise illegal snooping on them.

- **Fair Competition:** This is another interesting Attribute that requires the Trust Framework to comply with the antitrust laws that prohibit monopolising

a certain market. That is, any deployed Trust Framework should avoid practices that lead to vendor lock-in such as storing Users' Data using a proprietary format that cannot be processed by other competing Trust Frameworks or by preventing Users, or making it difficult for them, to move their Data to another competitor. While some software experts argue that the antitrust laws are irrelevant to the relatively new software markets, Katz and Shapiro argue against this idea with a detailed comparative study supporting their position [34]. In his famous Trusted Computing FAQs, Anderson describes some scary scenarios where few companies could control a singular Trust Framework that would enable them to abuse their powers to track people and control everything they produce digitally whenever it goes against the norms or the totalitarian laws that are followed by those companies [29]. For that, it is vital to distribute the Trust Management power among many competing Trust Frameworks that are governed by public and private bodies to insure proper usage of the powers that are associated with managing such Frameworks as well as to insure the existence of a safe resort for any oppressed minorities or individuals who cannot maintain their digital archive legally in totalitarian state-controlled Trust Frameworks.

- **Digital Rights Management:** It is true that the power abuse, in the form of oppressing freedom of speech, that is associated with the development of Trusted Computing is an undesirable side effect [29]. Nevertheless, the main motivation that ignites its development is the desire to protect the digital media from illegal pirating that costs businesses billions of dollars and undermines their ability to flourish [35]. For this sake, the developers of any Trust Framework should put at the heart of their design measures and technologies to minimise digital pirating to maintain the Trust of the Service Providers that the Users would respect their Digital Rights. In addition to the efforts of the Trusted Computing Group in trying to enforce Digital Rights Management, or simply DRM, a plethora of technologies such as watermarking and digital signatures are introduced in the Usage Control domain, or simply UCON, as we illustrate in Subsection 3.1.2.

2.7 Properties of the Trust Flexibility Requirement

While it is vital for any Trust Framework to enforce the terms of any agreed upon Trust contract among its Stakeholders, it is equally crucial for such Frameworks to be flexible enough to accommodate the differing needs of their Stakeholders in any possible context. That is, a flexible Trust Framework is a practical Framework that does not constrain its Stakeholders to a rigid implementation deploying only a limited set of features, lacking essential support, and unreliable due to the lack of sufficient software testing or adhering to the well-established standards. Further, a flexible Trust Framework is a Framework that enables its Stakeholders to customize their contracts based on the context of the desired transaction and the level of importance of the exchanged Data in the transaction. Finally, a flexible Trust Framework is a Framework that enables its Stakeholders to delegate some of their roles to other human or even digital assistants whenever they need to in a smooth, efficient, and secure manner. For that, we discuss in the following Subsections these Trust Flexibility Requirement Properties, as they appear in Figure 2.2:

- Practicality
- Customizability

2.7.1 *Practicality*

This Property of the Trust Flexibility Requirement is about ensuring that the Trust Framework is practical for its Stakeholders to deploy and use in their desired transactions. That is, any Trust Framework claiming to hold this Property must have a stable deployed implementation that support the main Trust Properties. Further, there must be reasonable documentation and support options to aid the Stakeholders in starting to use the Trust Framework and to troubleshoot any problems that may arise later on. Finally, a practical Trust Framework should be reliable to govern the Trust transactions among its Stakeholders by undergoing sufficient tests and adhering to well-established industry standards.

- **Documentation & Support:** This Attribute is about providing some essential documentation for the potential Stakeholders to start using the system as well as some support options to aid them in troubleshooting any future problems. Plus, this Attribute is about providing continuous updates and patches to existing problems because no software is immune to errors and, hence, there must be active developers to maintain the Trust Framework and keep it safe and Trusted. Of course, if the Deployment Attribute is not established, this Attribute would be of little importance.
- **Deployment:** This Attribute is about deploying a stable implementation of the Trust Framework in the real-life for Stakeholders to use. While there is a plethora of academic projects, prototypes, and beta versions of proposed Trust Frameworks, most of these proposals lack real-life deployments as can be seen in Subsection 3.2.2. That is due to the excessive amount of the required resources needed to carefully design a good Framework and, then, implementing such a design in a complete manner. Such resources are not available to individual developers or even academic units. Rather, such efforts would normally require state and industry sponsorship and support.
- **Reliability:** This Attribute is about applying essential testing to the implementation of the Trust Framework to make sure that it does what it is supposed to do in the normal, the special, and the unlikely use cases. That is, the testers should put on mind the inexperienced Users who may not follow the instructions or the Users who may input the wrong type of inputs whether intentionally, for malicious reasons, or not. The testers should also consider all the contexts where the Framework would operate let it be in a congested unreliable Network or among Stakeholders utilising different interfaces or APIs. Following well-established standardisation while implementing the Trust Framework as well as obtaining some form of testing certification confirming the quality of the implemented Framework are some forms of achieving this Attribute, see Section 2.8 for more details.

2.7.2 Customizability

This Property is about catering to the differing needs of the Trust Stakeholders based on the different Trust transactions scenarios and importance of the executed transactions and the exchanged Data within those transactions. It is crucial for the Trust Stakeholders to be able to engage in Trust contracts negotiations to come up with suitable contracts for the specific context of their agreement. Nevertheless, the ability to easily classify the Stakeholders' Data beforehand would facilitate the negotiation process as every party would be able to clearly distinguish between each Data type and how it should be handled internally once acquired.

- **Data Classification:** This Attribute is about enabling each Trust Stakeholder to classify its own Data. Whereas increased Security could minimize the Data breaches incidents, it would negatively affect the User experience and, hence, proper Data classification is needed to distinguish critical Data that should be treated with higher Security measures [36]. Such classification does not need to be binary. Rather, the Trust Framework may have a scale to rate the importance of each datum with a corresponding set of extra Security and Privacy measures attached to each level on that scale. In all cases, the Personally Identifiable Information, PII, are normally of higher importance than the Data sets that are generated by the owners of those PIIs. That is because the Data sets owners' Privacy could be exploited if their PIIs are not protected properly, see Subsection 2.6.2 for more details. For that, we present this basic Data classification scheme which could be further fine-grained by any Trust Framework:

- **PII Credentials:** These are like digital signatures or tokens that are associated with a single identity as described by Linn in [37]. Email address, home address, full name, phone number, and credit card number are few examples.
- **Data Sets:** These are larger Data sets that are owned by an identity. Digital documents, media files, written emails, or comments on social media are few examples.

- **Negotiation:** In real-life transactions, a one-template contract does not suite everyone. Digital transactions are no exception. If a company is to process a set of top confidential Data in a cloud environment, for example, it would require high Security Measures that may affect the QoS delivered. The other cloud tenants might view such requirements as hostile and absolutely unnecessary for their needs. Hence, it is essential for any Trust Framework to have flexible negotiation mechanisms to create tailor made contracts that suits the needs of every individual client. Such negotiation is usually costly [27]. The proposal of Dragoni and Massacci is a preliminary step towards optimizing the negotiation cost [38].
- **Delegation of Authority:** This Attribute is about giving proper access control to other human or digital assistants to do some tasks on behalf of one of the Trust Stakeholders. That is an essential Property in many real-life use-cases. For example, when a businessman wish to delegate an online Service Provider to sell or purchase stocks on his behalf whenever the prices reach pre-set thresholds. Another example is when an elderly woman wish to delegate her grandson to reserve medical appointments or to discuss her health status with the medical staff on her behalf. Proper delegation requires fine-grained access control where the delegate would not get greater authority and power than what the delegator intended to give. In addition, proper delegation should be easy to handle, revoke, and distribute among different delegates with different set of delegated authorities. There are many Trust Frameworks that address this issue such as the TAS3 and the OAuth 2.0 Frameworks, see Section 2.8 and Subsections 3.1.3 and 3.1.7 for more details.

2.8 Trust Measures Spectrum

The notion of Soft Trust, which was introduced by Rasmusson et al. [9], called the Trust status that could be achieved by the ratings and views of the digital social communities as Soft Trust Measures. In contrast, we would name the Trust status

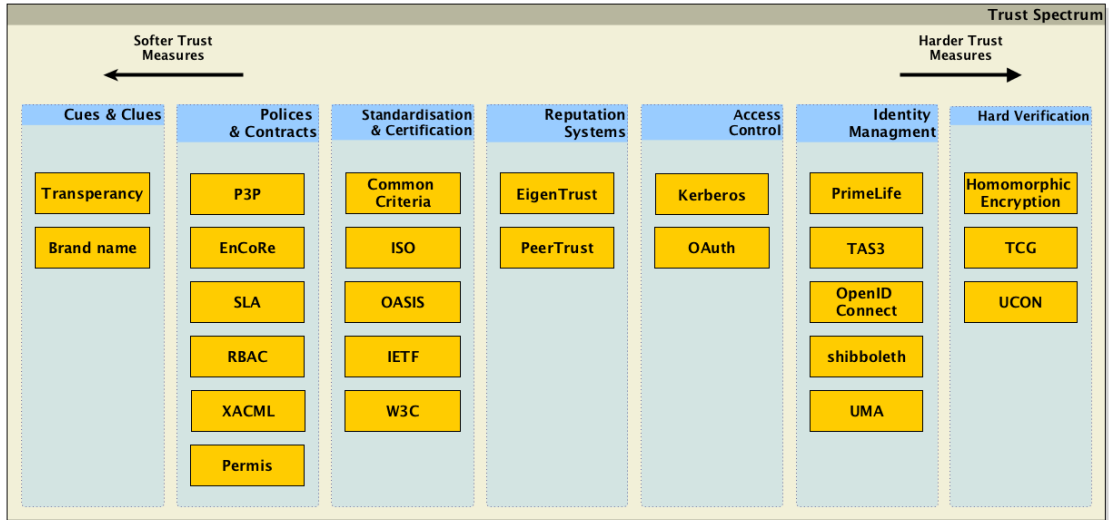


FIGURE 2.3: Trust Measures Spectrum

that could be achieved by utilising accurate proofs asserting whether a certain Trust party is conforming to the Trust contract as the Hard Trust Measures as suggested by Dragoni [27]. In between the Soft Trust Measures and the Hard Trust Measures is a spectrum of different Measures that we classify in this Section. Figure 2.3 illustrates the Trust Measures Spectrum per our classification with seven main Trust Measures Levels.

Before briefly commenting on each Trust Measure Level, it is important to note that classifying a certain Measure as a Soft Measure does not mean that it is worse than a harder Measure, and vice versa. In fact, any Trust Framework should incorporate a good balance of Soft and Hard Measures. Moreover, it should be noted that many of the evaluated Frameworks deploy utilities that would incorporate more than one Trust Measures Levels as we show in Section 3.1.

- **Cues and Clues:** This is the softest Measure Level in the Trust Measures Spectrum. In this level, the Service Provider initiates Trust with his clients by means of long established reputation, brand name, and transparency regarding the deployed Security and Privacy measures and polices. Despite lacking technical evidence, these measures have positive and significant effects on the ordinary Users' Trust status [16, 39]. However, the hardly established Trust is very easy

to lose due to a major Security or Privacy breach [16]. Such incidents are very likely to happen if no proper Harder Trust Measures Levels are deployed by the Trust Framework.

- **Policies and Contracts:** This is a little bit harder Trust Measure level as the Service Provider provides its Users with official guarantees in the form of legal contracts like a Service Level Agreement, SLA, which is a plain textual legal agreement. Despite its guarantees, it is hard in many situations to translate its promises into real actions in the Framework as the case, for example, in the cloud-computing field [15]. Another approach is to use the P3P, which is a machine understandable language to enforce Privacy Requirements of legal agreements [40]. One project that took a step forward in this Trust Measures Level is EnCoRe that aims to develop a tool that maps SLA abstract agreements to machine understandable languages like P3P [41]. Another approach to formally define access rights is the classic Role Based Access Control, RBAC, where access to certain Data is restricted to Users with specific roles [42]. The literature is rich with policy languages to map RBAC roles to machine understandable forms. Among the highly cited and utilized such languages are XACML [43] and PERSIM [44]. Nevertheless, without proper strong Enforcement, all the generated policies in this Trust Measures Level would be useless except in the case of a legal dispute which is normally expensive and poses the hard challenge of proving any agreements' breach.
- **Standardisation and Certification:** In this Trust Measures Level, the Service Provider proves his competence and willingness to adhere to any signed contract by showing a quality certificate that is obtained from an official Trusted Authority. One of the most notable certification system that is currently deployed and sponsored by many international governments is the Common Criteria Catalogue [45]. There are seven levels of Security evaluations ranging from the most basic, EAL 1, to the most comprehensive, EAL 7. A full source code evaluation is done only at the highest evaluation level, which is also the most expensive one

to implement. In addition, it is usually hard to compare the results of two certified products, which makes it hard and confusing for a client to decide on which provider to deal with based on their certifications [12]. Other possible forms of certification could be done by developing protocols by dedicated groups of developers governed under the umbrella of a standardisation body like ISO [46], IETF [47], W3C [48], or OASIS [49] as examples. A formally developed protocol should be generally safer to use since it is supposed to be thoroughly inspected by respected participants coming from different entities. However, the developers are, at the end, human beings that are prone to be affected by politics, personal interests, or simply human mistakes. An example of a controversial standard is the IETF standard OAuth 2.0 where the main editor resigned from his position claiming that the group have made the new version of the protocol so loose that it is impractical for small firms and Users to make correct use of it [50].

- **Reputation Systems:** This is the middle point in the Trust Measures Spectrum. In this Trust Measures Level, the social power is exploited to generate ratings and recommendations [13, 51]. In addition, many proposals and projects utilized a form of evidence gathering from previous interactions to generate more comparable and Trusted ratings [27]. Despite the effective power of social ratings, there are serious challenges that face the advocates of this line of research. Among those challenges, per [27], are to deal with evaluating the new entrants to the systems, which is partly examined in [52], to deal with human subjective ratings, and to deal with the malicious nodes who attack the integrity of the ratings [26]. It is worth mentioning that the highly cited Trusted P2P projects, EigenTrust [53] and PeerTrust [54], are not implemented for real-life deployment yet and, hence, cannot be evaluated for their effectiveness [27].
- **Access Control:** This harder Trust Measures Level is mainly concerned about restricting access to Users' Data or Credentials to only authenticated and authorised entities. In other words, it tries to implement the Enforcement measures of the Sticky Policies that are found in the above-mentioned policies and contracts

Trust Measures Level. One of the oldest, and the most reliable, access control protocols is the classic Kerberos [55] where mutual client and server authentication takes place through secure communication channels by incorporating a Trusted third-party who knows both the server and the clients. Another more recent example is the OAuth 2.0 protocol, which tries to make it easy for a User to give certain entities access rights to a limited set of his Data or Authorities [56]. While this Trust Measures Level is effective in simple access scenarios, it is not sufficient in complex scenarios where, for example, the Data owner requires unlinkability, anonymity, pseudonymity, or Data protection after release.

- **Identity Management:** In this Trust Measures Level, the main focus is on managing the Users' digital personalities and Credentials. That is, providing the set of tools and usable mechanisms to create contextual profiles, delegate the use of certain Credentials, support Attribute based authentication, and anonymise the User personality whenever necessary. Some of the worth mentioning projects that tries to implement those ideas are PrimeLife [57], TAS3 [3], OpenID Connect [7], Shibboleth [58], and UMA [59] projects, see Section 3.1 for more detailed analysis of these Frameworks. In fact, the TAS3 and PrimeLife are inactive academic projects and the Shibboleth is concerned with a limited set of inter-institutional authentication. In contrast, the OpenID Connect and UMA are active projects that are concerned with more mainstream identity management scenarios. Per Maler, one of the key developers of the UMA project, the new vision for mainstream access control is illustrated in Figure 2.4. In that Figure, we can see that the trio of the OpenID Connect, UMA, and OAuth 2.0 could be integrated to support a wide range of modern access control scenarios. That is, the task of **Authentication** would be taken care by the OpenID Connect protocol, which manages User's identity claims and provides Single Sign-On, SSO. Further, the OpenID Connect provides identity tokens, after the User consent to do so, enabling requesting third parties to access a set scope of the managed User's identity claims such as name and email address. The OAuth 2.0 protocol would take care of securely **Authorising** apps and Users, by confirming the validity of the access tokens they provide. Finally, the UMA protocol would

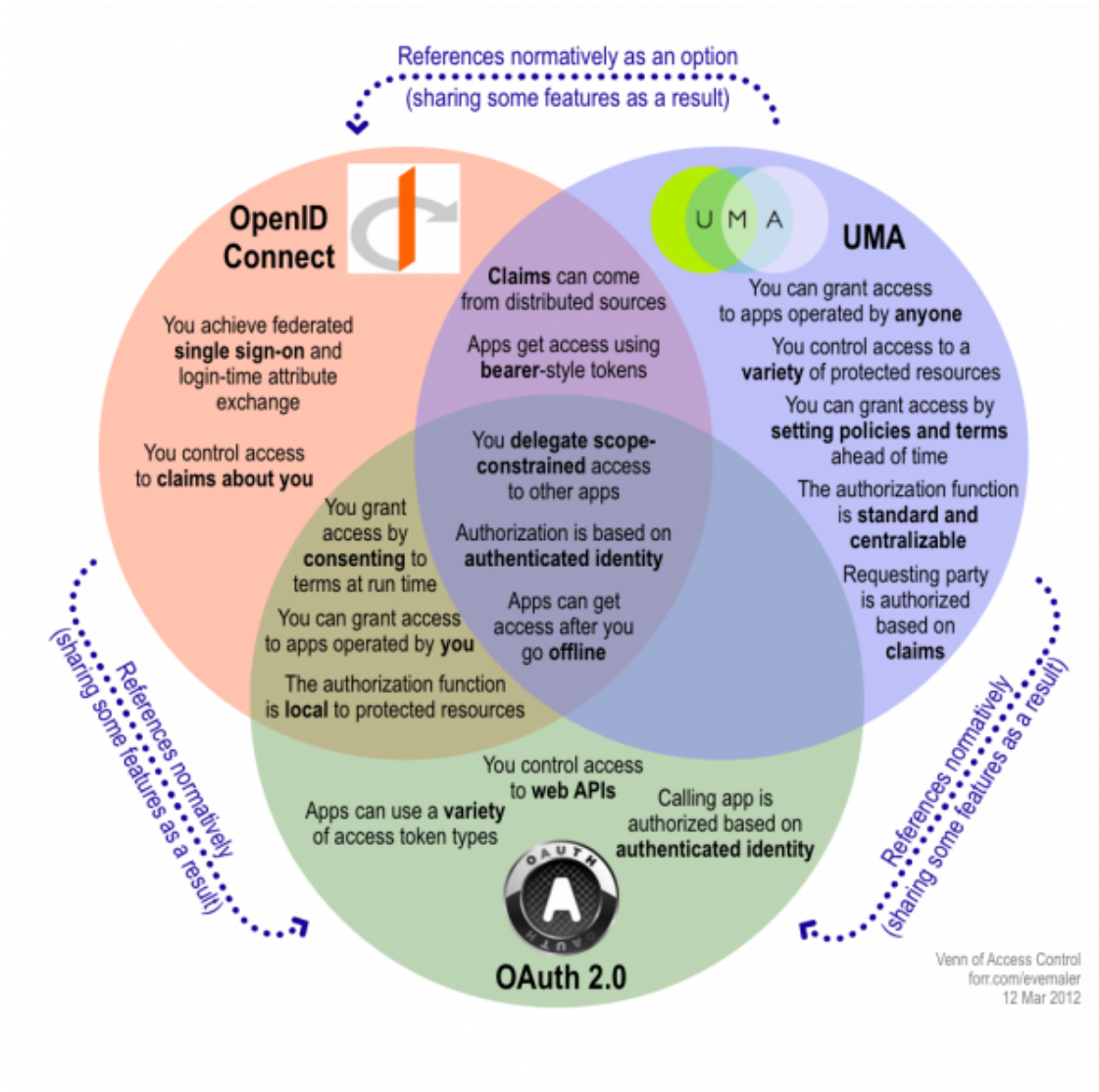


FIGURE 2.4: A New Venn Of Access Control For The API Economy [1]

take care of the **Delegation** task where a User would categorize his Data and Credential into different scopes, which the User give consent for other Users or apps to access in the negotiation process during the authentication process that is carried out by the OpenID Connect protocol [1].

- **Hard Verification:** This is the hardest level within the Trust Measures Spectrum where the main focus here is to achieve the highest form of certainty despite the high QoS cost. In other words, it can support the Enforcement of the previous layers rules even after Data release. The list starts with a strong cryptographic mechanism called Homomorphic encryption [60] which is,

despite being introduced in 1986, still impractical because of its high processing cost. The work of Gentry provides significant optimization to the Homomorphic computation cost [61], but it is still expensive to be implemented in real life applications. The TPM technologies that are being developed by the Trusted Computing Group, TCG, is aiming to generate Hard Trust by means of software and hardware remote attestation to ensure that every component is behaving as expected [62]. Once a component, whether hardware or software, misbehave, it would transparently show its status to the concerned parties. Despite its potential benefit, it raises many concerns among the ordinary Users about their Privacy and control over their own machine [29]. However, proper legislation would eliminate such concerns and fears [28]. It is worth mentioning at this point that relying solely on TPM technologies would not solve all our Security and Trust issues; Rather, it is a building block toward a proper Trust Framework [63].

Chapter 3. The Need for Continuous Trust Management Frameworks

In Chapter 2, we answer fundamental questions regarding the concepts of Digital Trust and Digital Trust Management Frameworks by surveying the literature. In particular, we list a set of main Challenges currently facing Trust Management Frameworks, see Section 2.4, today along with a list of essential Trust Requirements, see Section 2.5 that should be fulfilled by any Trust Management Framework to counter the current Challenges. Further, we present a Trust Measures Spectrum, see Section 2.8, that categorizes the current Trust Management Frameworks per the type of Measures they deploy to establish and offer Digital Trust.

In this Chapter, we present a collection of some of the existing Trust Management Frameworks that we picked based on the breadth of the Trust Spectrum they cover and the variety of the Trust Attributes they implement. Following that, we explore here to what extent those Trust Frameworks fulfilled the Trust Requirements of Section 2.5 and by using what Measures from the Trust Measures Spectrum of Section 2.8. We present the results of our comparisons in tabular format to ease comprehending. These results are followed by a discussion of the strengths and weaknesses of the investigated Trust Frameworks.

Our investigation will show that while the investigated Trust Frameworks excel at fulfilling traditional Security Attributes, they fail at providing Continuous Data Control because there are limited, under-researched, measures to protect the Trust Stakeholders Data after they release it to other parties. This fundamental weakness restrains the current Trust Frameworks from providing Continuous Trust for their Trust Stakeholders and makes it impossible to fulfil all the Legal Authorities Trust Attributes. In addition, the lack of Continuous Data Control makes it less significant to invest in

strengthening the current Customizability Trust Attributes. For that, we argue by the end of this Chapter for the importance of developing a Continuous Trust Management Framework that is designed and implemented to provide the missing Continuous Data Control.

3.1 Current Trust Frameworks at a Glance

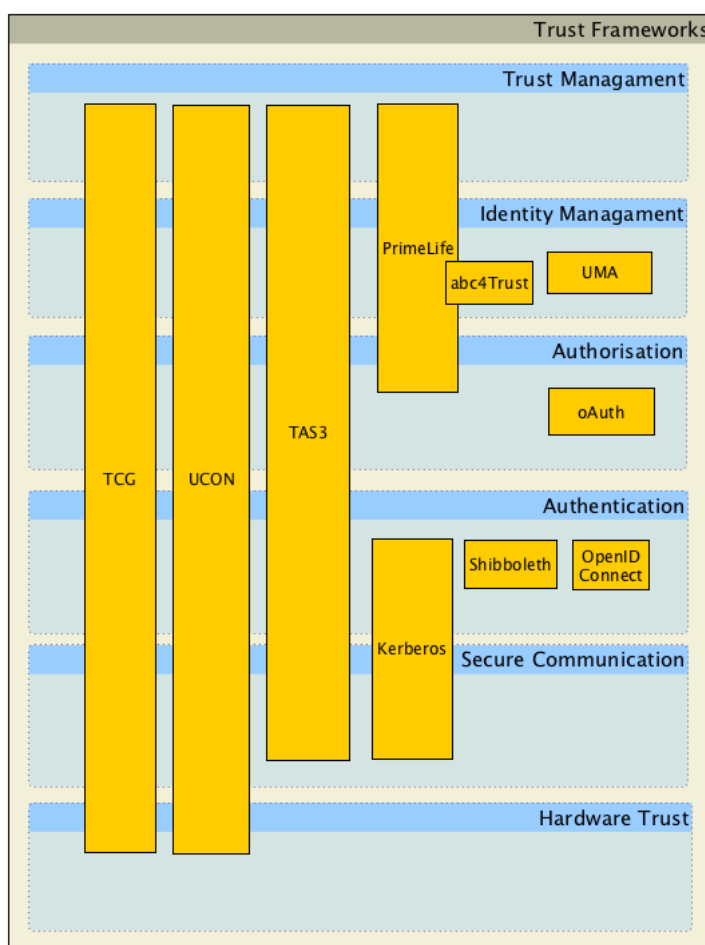


FIGURE 3.1: The Current Picture of The Trust Frameworks/Protocols Literature

Unlike the late 1980's where a literature survey for Trust Frameworks may return a set of academic projects to secure the Data communication channels, the literature picture today is more vibrant and complicated. In the current picture of the Trust literature, the hot spot is not the communication channel; rather, more attention is currently paid to Users' Privacy, Data protection, and access management rights. While some Trust

solutions are well-established protocols; many are still in their infancy as academic projects or in their adolescence as draft protocols. To add more complexity to the picture, some solutions focus on specific Trust issues like authorisation or identity management for example while many recent projects take a more holistic approach toward implementing the essential Trust Requirements' Properties and Attributes, see Section 2.5.

At the moment, the topic of Trust Management Frameworks is not largely studied as a unit. Instead, each particular Trust issue, or a group of closely-related Trust issues, are studied alone. For that, there is no unified view of how the Trust Stack should look like. To capture the current picture of the Trust Frameworks and how it is evolving, we show our Trust Stack view in Figure 3.1, where we put the hardware and the basic, but essential, Security features at the bottom of the stack while putting the more advanced Trust layers that are concerned with Privacy, Trust ranking, and Trust dissemination on the upper layers of the stack.

In Figure 3.1, we list few representative Trust Frameworks that were selected from different pools and categories to sample the current residents in the Trust Frameworks literature. In that figure, we list the evaluated Frameworks in the layers where they mostly operate, for more details, see Section 4.1. The selected Trust Frameworks have been carefully analysed to extract different sets of comparable features and trends. In the following Subsections, the analysed Trust Frameworks are briefly introduced. In Section 3.2, we compare these Trust Frameworks on the aspects of their support to the Attributes of both the Trust Enforcement Properties and the Trust Flexibility Properties as well we their utilisation of the different Trust Measures Levels.

3.1.1 TCG

The Trusted Computing Group, TCG, approach towards Trusted computing is to root all the Trust chains to a Trusted hardware chip, called Trusted Platform Module, TPM, that has many built-in Security features like storing sensitive Data and generating secret keys [64]. Due to its generic nature, TCG standards could be used

in a variety of fields like: risk assessment, e-commerce, computing assets management, Security monitoring and emergency response [64]. As of the time of writing this thesis, the TCG group is active and running with an estimated number of around 2B devices deployed with TPM chips [65].

To grasp the technical image of TCG, Lee-Thorp [66] mentioned that an ordinary TCG platform is usually made of the following main technical components:

- **Trusted Platform Module, TPM:** this is a tamper resistant chip [64] that is capable of performing Security tasks like keys and passwords management as well as encryption/decryption tasks without the need to depend on untrusted communication channels or software layers [67].
- **Root of Trust for Measurement, RTM:** the RTM is a computing engine that is capable of measuring and securely storing integrity metrics. The initiation of the RTM is normally done by a pre-Bios Boot Block, BBB, that is called CRTM. CRTM is also expected to be tamper resistant and only execute Trusted pre-authorised software [66].
- **The Software Stack, TSS:** this stack is an interface to access the functions of the TPM to extend the hardware Trust to the application layer. Among its features, TSS include interfaces for existing crypto APIs which extend TPM support for applications using those APIs [67].

Based on TCG's Trust features and technical components, it is predicted that TCG standards would enable protected storage facilities in the short run, improve access management policies based on the running software status in the medium run, and establish strong system integrity measurement based on the reporting and attestation features in the long run [68]. The long-run vision should be enabled by the integrity metrics reporting functions, which is also known as remote attestation [66]. The purpose of the remote attestation is to verify to an external challenger the integrity metrics of the running system. The current attestation model is designed around the idea of computing the hash values of the Trusted software or OS and, then, comparing

those securely registered hashes with newly computed hashes at the boot time. If the pre-computed hashes matches the newly calculated hashes, the TPM will vouch for the integrity of the running software. In order to protect the device owners' Privacy, it is possible to use Direct Anonymous Attestation protocol, DAA. This protocol basically assumes a Trusted Certification Authority, CA, that knows all the Endorsement Keys, EK, of the valid TPM machines. Then, if a TPM owner wishes to prove attaining certain properties without compromising their anonymity, he could contact the CA along with a newly generated key called Attestation Identity Key, AIK, to be verified against the User's EK. If the transaction success, the CA would sign the AIK and return it to the User who can then forwarded it to the verifier. [69].

If the TCG protocols got widely adopted, we could expect a revolutionary trusted computing experience. Nevertheless, there are still many issues and research challenges facing the TCG ultimate goal of fully trusted computing. Among those issues are:

- *Trusting Roots of Trust* may be achieved through a variety of ways but is anticipated to include technical evaluation by competent experts [64]. "That is said and by looking at the TCG documentation road map [64], we see that the experts are assumed to be the Common Criteria body. Per the Common Criteria manual [70], code level verification is done only at the highest assurance level testing, EAL7. However, the Common Criteria website [71] indicates that the three certified trusted computing products have got EAL4 certificates. For Security sensitive operations, is it sensible to root all the computing Trust to unverified source code?
- TPM is assumed to be tamper resistant but, in practice, it is not. As mentioned by Sadeghi, the current practice is to connect TPM chips to the I/O system using unprotected channels that can be easily compromised. A proposed solution is to use cryptographically protected communication channels for TPM chips [63]. To further demonstrate the shortcomings of the current TPM chips, Christopher Tarnovsky showed how to access the non-volatile memory containing the Users Data in the Infineon's TPM in Black Hat 2010 [72].

- In the current TCG architecture, TPM could become a bottleneck to the system performance. Plus, it does not scale pretty well in open computing systems [66].
- As mentioned earlier, the attestation model assumes static hash values while, in practice, those values can be easily modified during the run time by exploiting Security bugs or swapping the memory, for example [63]. Moreover, it is almost impossible to Trust the code of nowadays complex software which means that verifying their hash values does not make great sense. The attestation metrics for software are low level metrics like the configuration file or software image while the more important property would be the software semantics and behaviour [66]. Moreover, even if it is attested that a platform is genuine, the vendor usually does not guarantee the behaviour of the platform. Another limitation of the current attestation model is the fact that it cannot guarantee capabilities or system functionality. Lastly, TCG attestation currently doesn't refer to a single running instance of a given software; rather, it refers to a software entity in general. As a result, a better Trustworthiness evaluation methodology is needed.
- Per Pearson, without proper legislation, a huge spectrum of commercial abuse to the consumers Privacy and freedom could result by the TCG standards [28]. That is due to the potential power of the remote attestation that could, for example, enable some vendors to sell software or media that could be read for a limited number of times or consumed in certain locations only. More threats and concerns are described in the 'Trusted Computing' Frequently Asked Questions [29]. Proper legalisation is essential to promote the use of TCG standards in a wide scale.

3.1.2 UCON

The Usage Control, or simply UCON, is an approach focused on access management that was introduced by J. Park in 2002 [2, 73] and then formally modelled by X. Zhang [74]. The aim of UCON is to provide a unified Framework that can provide

dynamic fine-grained access control to digital resources before and after release. That is, a UCON system should be able to offer client-side reference monitor to enforce the dynamic access policy which is dependent on the client usage in contrast to the traditional access control approaches like Mandatory Access Control, MAC, or Role Based Access Control, RBAC which are capable of only providing static access policies through server-side reference monitors that cannot enforce any access rules after granting access to the client. In other words, the UCON approach ”encompasses traditional access control, Trust Management, and digital rights management and goes beyond them in its definition and scope [2].” Since its first publishing, UCON attracted a lot of academic attention and its development became an active research area [75]. Nevertheless, there is not a formal research group or standardisation body that supports this approach, as far as the authors know.

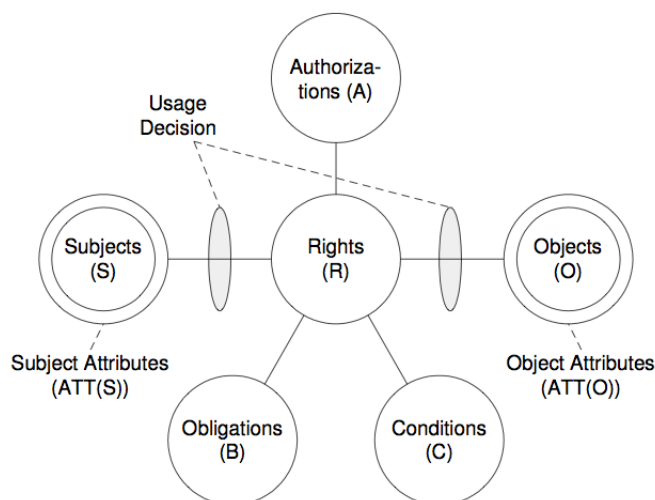


FIGURE 3.2: UCON Model Components [2]

The UCON high-level components model, based on the work of J. Park, is shown in Figure 3.2. This model consists of subjects, having certain attributes, trying to get rights to access objects, or some of its attributes, that are protected by the UCON Framework. The UCON model differs from the traditional access protocols by utilising the authorisations, obligations, and conditions components, or simply ABC, where the traditional protocols would rely on the authorisation component solely in making access decisions. The authorisation component encapsulates who has rights to access

what while the obligation component takes care of checking if any required obligation by the requester is fulfilled, like filling up a disclaimer form, and the conditions component is responsible for making sure that any environmental or system-oriented, like the presence of an anti-virus, is fulfilled. Finally, the UCON model realise two important properties that are missed in the traditional systems: continuity and mutability. The first means that the policy Enforcement should continue even after granting access rights while the second means that certain attributes, like number of usages, could be automatically updated instead of manual attributes update like in the traditional systems case. Utilising the ABC components, continuity of usage policies, and mutable attributes, a UCON system can maintain two usage decisions points: the first point to grant the initial access right and the second to update the access right based on the usage and deployed policy [2].

A survey covering the active research in developing the UCON protocol showed many different realisation and variations on adopting the UCON model [75]. For the purpose of this comparative study, the reported features from the many different research papers to develop UCON models which are presented in the Lazouski et al. survey [75] are treated in this study as if they are a single UCON Framework implementation for simplicity reasons. In terms of UCON architecture, the key component is called the Reference monitor which is like a gateway that receives any request to access digital Data. Each request would be evaluated by a Policy Decision Point, PDP, and its decision would be enforced by a Policy Enforcement Point, PEP. The reference monitor could be located in the server-side, the client-side, or in both locations. In the hybrid approach, the server-side part of the reference monitor could handle Data protection before release while the client-side part could take care of enforcing its usage policy after release.

Looking at the PEP in more details, there are many Enforcement methods that were surveyed by Lazouski et al. [75]. One of the reported methods is digital watermarking which is basically about injecting an invisible piece of Data that is attached to the protected Data object. The purpose of the digital watermarking is to track the Data distribution and whether the object is distributed per the agreed upon contract

or not. Another Enforcement approach is the digital container where a Data object is encrypted and cannot be accessed, even after release, unless the Virtual Machine, which contains the PEP module, decrypt it. Of course, this VM should be implemented in a tamper-resistance way or it would be useless. Tamper resistance VM could be achieved by code obfuscation, to prevent altering the code. Another approach towards tamper resistant VM is called Model-Based Behavioural Attestation, MBA [76]. MBA utilizes low level hardware attestation techniques, such as TPM chips, to facilitate a higher-level attestation layer that evaluates a Data consumer Trustworthiness based on her behaviour [76].

The UCON approach towards Trust Management covers many important aspects of it. In fact, it is among the few Trust Management approaches that provide effective proposals to protect digital Data after release. Nevertheless, this approach still faces many issues and obstacles that needs to be tackled. First of all, there is no standardisation body or even an academic research group that takes care of generating a unified standard or protocol for UCON. Without an agreed upon standard, UCON would remain an inspiring research theme that is hardly applicable to real-life computing. Moreover, despite the proof of concepts implementations that are scattered in the literature, there are not full implementations that could be evaluated for their reliability and effectiveness. That is important because when it comes to policy Enforcement after Data release, by means of VMs, the tamper resistance feature is a conceptual idea that is challenging to implement in practice. When it comes to tamper resistance, code obfuscation approach requires using closed source software, otherwise, the malicious client may replace the obfuscated VM with a fresh copy of the open source software. Closed source software may provide Security but never provide Trust since the fear of a malicious code developer sharing the source code with malicious clients. Hence, the implementer would be left with hardware tamper resistance approach with all the challenges currently facing the TCG attestation, see Subsection 3.1.1.

3.1.3 TAS3

The Trusted Architecture for Securely Shared Services, TAS3, project is a EU funded project that was concerned with developing a generic Trust architecture for Web Services computing. As mentioned by Kellomaki, the main objectives of this project are to fulfil the requirements of the complex and heterogeneous business process, to empower ordinary Users with User-centric and dynamic access management policies, and to secure the communication channels that transmit PII Data among different entities [3].

Per Kellomaki, Trust should be technically enforced in the TAS3 architecture [3]. In case that was not possible at some situations, a Legal Framework that is proposed by Alhadeff and Alsenoy should be used as a last resort [77]. Some of the design objectives of the TAS3 architecture, per [3], are:

- Empower Users with the ability to manage how their PII Data and attributes are used and disseminated.
- Provide useful, and accessible by Users, distributed auditing system.
- Employ a binding Legal Framework to complement the technical enforced Trust.
- Utilize a set of Trusted third-parties to manage and enforce Trust related tasks like authorisation and keys management.
- Deploy strong encryption and Privacy preserving protocols.
- Develop sticky-policies to cryptographically attach access policies to Data. Those policies should be empowered by policy Enforcement infrastructure.
- Enable auditing and quality assurance entities to test whether online services do comply with their specifications.

Per Alhadeff, the TAS3 project views the Trust ecosystem as a collaboration among different entities to enable the TAS3 architecture [77]. Such collaborating entities

would form what is called a Trusted Network, TN. The TN would involve the following main Stakeholders:

- **Data Subjects, or simply TAS3 End-Users:** These End-Users include Service Providers and service requesters of either computing application services or Trusted Third Party services like certification agencies or reputation engines.
- **TAS3 Governance Entities:** Those entities may include a centralised Trust Guarantor and/or a TN Governing Board that could be made of the TN Stakeholders in addition to some Legal Authorities.

It is also mentioned by Alhadef that the ideal situation to bootstrap a TN is to have a central Trusted and already respected authority to anchor the TN. Otherwise, a consortium of smaller Trusted entities should assume that rule [77].

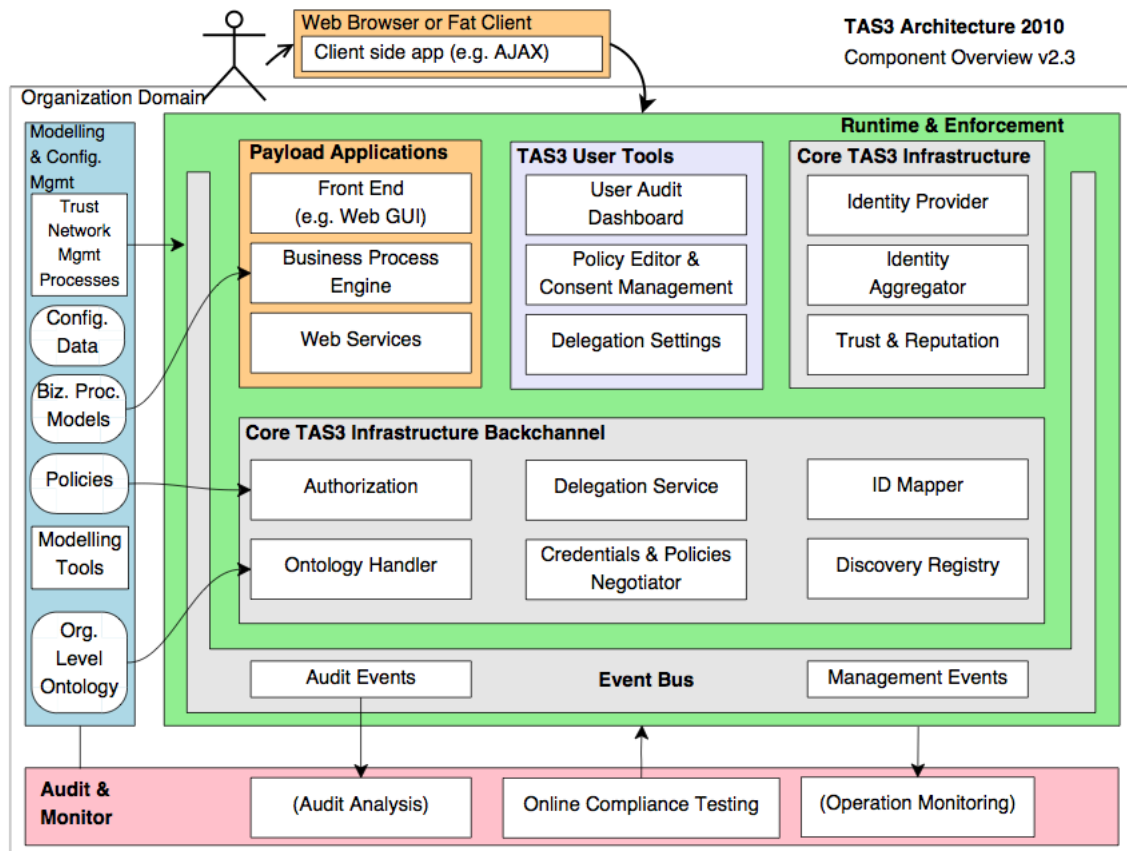


FIGURE 3.3: A Components Overview for the TAS3 Architecture [3]

To get a deeper understanding of how TAS3 works, Figure 3.3 shows a detailed view of the architecture's components. Some of the main components in that figure per Kellomaki [3] include:

- Authorisation services.
- Authentication services.
- Privacy preserving services: to generate pseudonymous identities and minimise the identities and attributes linkability.
- Trust negotiation services: to decide if the other end of the transaction is Trustworthy enough to start a dialogue.
- Secure business process management services: to help business services to operate and get dynamically updated in a secure manner.
- Delegation services: to enable a User to delegate Credentials to another User or agent.
- Discovery services.
- Trusted registries: to track all services in the TN that provide services conforming to the TAS3 specifications.
- Attribute authorities: to vouch that specific Users possess certain attributes.
- Secure repository services: to store Users' attributes securely in accordance with their attached policies.
- Trust and reputation services.
- Secure audit services: to provide a tamper resistant log of all the transactions made within the TN boundaries for the use of legally authorised entities.
- On-line compliance testing services: to frequently and anonymously interact with all the services in the TN to ensure their compliance with their published specifications and policies.

As can be spotted in the above objectives list, Users' Privacy is a core component of the TAS3 architecture. Per Vandevenne et al., the TAS3 researchers do not think that Trust or Privacy are negotiable; rather, satisfying the access control policy by giving a required set of Credentials is what agents should negotiate. Moreover, The TAS3 negotiation module is designed to enable the User to prove to the Service Provider that it holds certain attributes without necessarily revealing all of those attributes [78]. In fact, each personal attribute may have an associated access policy, Sticky Policy, requiring the requesting server to fulfil certain promises or to possess certain qualities before the attribute get transferred. Similarly, the requesting server may not reveal all its policies to clients. Rather, some parts of the policy could be publicly accessed (like terms and conditions) while other parts like (which employees can access clients Data) is restricted to internal entities [78].

A possible implementation of the Sticky Policies, which Kellomaki describes, is to develop a language to express and exchange the Sticky Policies, named: TAS3 Simple Obligations Language, SOL. That language, or any other similar language, would attach an access control and retention policy for each Data item. When a requester asks for a specific Data item, he would send pledges regarding how he would treat the requested Data. The Requester Policy Enforcement Point, PEP, would check the received pledges against the Data attached obligations and, if they match, it would send the requested Data [79]. More details about the Sticky Policies and how we are using them can be found in Section 4.4.1.

The TAS3 project provides a great Framework that considers Users' Privacy at its core design. Despite being an inactive academic project, its outcomes could be utilized to develop a more robust practical Trust Network. Nevertheless, we observe many deficiencies like:

- The Sticky Policies are not enforced once the attributes are transferred from its owner to the requester. It is clearly said in the project's architecture document that Sticky Policies should be ideally attached to the attributes they protect by cryptographic means to prevent its disclosure unless with accordance with the policy terms [3]. However, it is also mentioned in that document that "this is a

difficult research problem and will be addressed in other TAS3 deliverables [3].” As far as we know, this issue was not addressed in any other deliverable. In that case, once the Data leaves its owner, there would be no technical guarantees about enforcing its obligations except for the reputation rankings, which cannot detect passive Data snooping or even some active manipulation where the Data owner cannot detect which requester have breached the exchanged Data policy. Of course, it is possible to rely on legal Enforcement but, again, if the Data owner cannot technically proof who breached his Data, the legal path would probably leads to nowhere. Utilising Some form of software attestation, similar to the TCG or UCON approaches, to launch the TN to ensure the integrity of the running Enforcement software would dramatically enhances the Trust level of and reduce the risk of colluding parties or even contract breaches.

- It is not clear from the documentation if it is possible for the Data owner to retrieve access to his Data if he found that the requester has breached the contract. But it was clearly described by Kellomaki that the Data owner can attach a retention policy using the SOL language [79]. Again, if software attestation is to be used, it would be possible to keep the Sticky Policy attached to the Data wherever it goes and, hence, it would be possible to implement a revocation mechanism.
- The online compliance testing services, which is part of the TAS3 architecture, is a good idea in concept. In practice, it was not clear in the documentation how the testing would be carried out if the required service needs payment. Since the testing request would be anonymous, it should behave like a normal User and pay for the required service. It could be that after the end of the transaction, the testing module reveals its identity and ask for repayment. In that scenario, the Service Provider could analyse the testing module pattern and, hence, could guess if an incoming request is a test or not which would break the testing algorithm. A better approach could be by asking each member of the TN to pay in advance a membership fee to cover the cost of testing.

- Another issue with the testing module is the fact that it cannot detect passive Data snooping. However, it could detect some forms of active Data theft if it uses a strategy like sending random attributes to only one entity and then monitor the Internet to check if the exchanged Data have been leaked or not. If yes, it would be easy to map the leaked Data to the only suspicious service provider and, hence, this provider could be removed from the Trust Network.

3.1.4 PrimeLife

The PrimeLife project is a EU funded project that is concerned with tackling the new Privacy challenges that were posed by the emerging web that formed online social communities, mashup applications, and lifelong storage with nearly unlimited storage [57].

Users' Privacy lies at the heart of the PrimeLife project. The PrimeLife researchers think that to protect Users' Privacy, Users should reveal the minimum required knowledge while the other end Data usage should be governed by access policies [57]. Moreover, as a mean to better control lifelong Privacy, hiding Data through identity management and User control is usually preferred over its disclosure as it is impractical to remove personal Data from other entities devices [57]. Per Pfizmann et al. [57], the PrimeLife Project acknowledges and tries to fulfil the following Privacy requirements, if needed by Users or legislators:

- **Privacy of attributes that leads to direct identifiability:** That includes both anonymity and pseudonymity.
- **Privacy of attributes that leads to indirect identifiability:** That includes Data confidentiality, Data storage and processing minimisation, and empowering Users with Privacy control tools.
- **Contextual integrity**

The PrimeLife project is an umbrella for many smaller projects that all aim to tackle the Privacy issues in different domains. Nevertheless, one mini-project that we consider as a Trust Framework is the PrimeLife Privacy-preserving access control system [80]. This proposed architecture has a generic SOA architecture to support the Privacy oriented Data handling that is proposed. Furthermore, it is independent from policy languages or specific deployment platforms to keep it open for specific implementations. To come up with this generic architecture, the developers came up with 39 general requirements to cover the different area of Privacy, authentication, PII access, and cross domain communication [80].

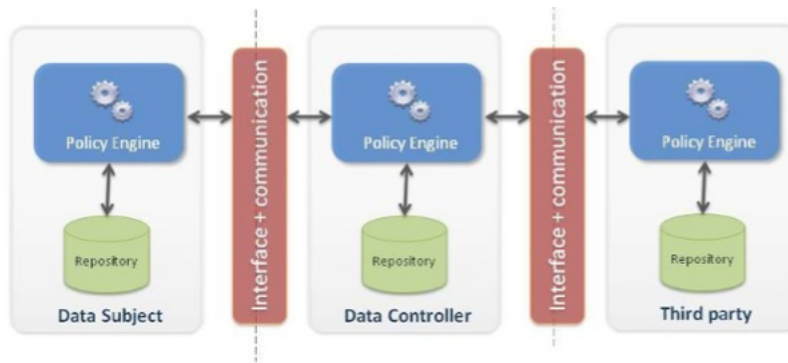


FIGURE 3.4: A High-Level Architecture for the PrimeLife Privacy Language [4]

The basic idea of this architecture is to match the Privacy policies governing handling the PII from the different providers and consumers with Sticky Policies that controls the release of the Data to only the authorised entities without leaving any linkable traces in the SoC sphere. For such a solution to work, the policy language should be as precise as possible to cover all the legal aspects. Examples of the currently deployed policy languages are XACML and P3P which, although providing fair enough functions to express the Privacy policies and to handle access control, are limited in dealing with Data handling and in their understandability by Users. PrimeLife project worked on extending the ontologies of those languages to address some of those raised issues. Trabelsi and Njeh describe the detailed architecture of the PrimeLife Police Language, PPL, which extend the XACML access control policy to enable Data Handling Laws Enforcement by means of encryption [4]. The general picture of this proposed architecture looks like Figure 3.4:

While PrimeLife project offered a well thought about Privacy-oriented architecture for Data exchange, there are many rooms for improvements in their proposal. First of all, despite the fact that PPL improved the way in which the currently deployed policy languages such as XACML and P3P handle access control, it should be noted that XACML, and any other language that is extending it like PPL, are concerned with how to attach the access control policies to the Data but not with how such policies are enforced at the end-points. For that reason, once the Data leaves its owner, there are no technical guarantees offered by the PrimeLife Privacy preserving architecture regarding how the Data would be treated after its release.

3.1.5 ABC4Trust

Per Camenisch et al., the main objectives of the Attribute-Based Credentials for Trust project, simply ABC4Trust, is to define the main components and Data artefacts in a common abstract architecture for all systems that are meant to implement Privacy-preserving Attribute-Based Credential systems [5]. While the Prime and PrimeLife projects showed that ABC systems provide good Privacy, they presented a limited number of demonstrators which means a gap between theory and practice that raises the need for the ABC4Trust project.

The general goals of the ABC4Trust project, per Camenisch et al. [5] are:

- To present a generic Framework that can accommodate the different Privacy-ABC systems by identifying the functional modules and producing suitable specifications for the Data objects, APIs, and protocols.
- To present clear criteria to compare the features of the different implementations of the proposed modules.
- To present reference implementations for each of the proposed modules.

Per [5], the main Privacy issues facing the identity management systems today are:

- The Service Provider knows all the Users' transactions history.

- Possible linkability across different domains.
- The identity proportionality is often violated (by collecting excessive number of personal attributes).

The proposed ABC4Trust architecture, per Camenisch et al., attempts to overcome the abovementioned issues [5]. This proposal does not only conform to the basic laws of Privacy like the principle of necessary processing (which became mandatory in countries like Germany’s for eID usage), but also open the door for future research to deploy stricter legalisation by means of:

- The wide variety of attested personal Credentials that would be possible to collect, thanks to ABC4Trust, without revealing the whole set (selective disclosure). Nevertheless, the wide deployment of such systems would introduce the challenge of establishing proper methods to process them legally.
- Currently, many Data controllers are needed to collect the personal Credentials and transactions logs for future inspections, if needed. The inspection feature offered by ABC4Trust would eliminate this need as the Credentials would be offered whenever needed by a credible inspection authority.
- The unlinkability feature would enable the Enforcement of purpose-binding requirements using the ABC4Trust architecture.

Among the goals of this architecture is to enable its Users to deploy the ABC4Trust features on top of the existing technologies such as WS-*, SAML, OpenID Connect, OAuth 2.0, and X.509. It was analysed by Camenisch et al. how this architecture could be integrated with the aforementioned technologies to overcome some of their Security and Privacy shortcomings [5].

The main Stakeholders in the ABC4Trust architecture are shown in Figure 3.5. In that figure, the User is the human client who tries to access resources protected by the verifier who demands certain Credentials to be provided by issuers who may specify some terms and conditions to enable certain inspectors to de-anonymise the Credential

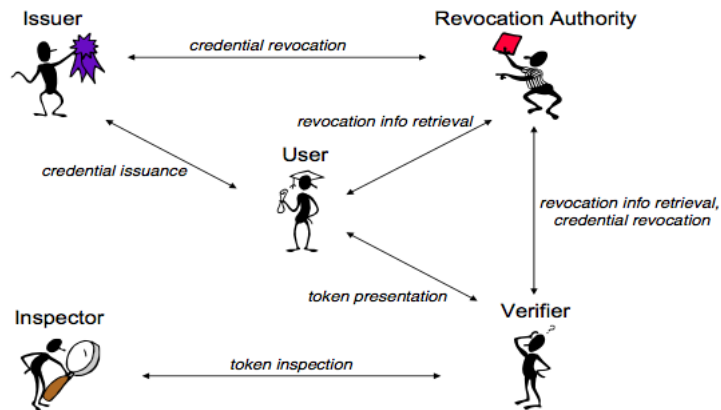


FIGURE 3.5: Architecture for Attribute-based Credential Technologies [5]

token or to enable a revocation authority to invalidate the use of the token with certain or all verifiers. The revocation authority could also be demanded by the User to disable the utilisation of his presented token to certain providers without affecting the usage of the same Credential with other verifiers. That is important so that a malicious verifier could not use the token for identity theft purposes. It is worth noting in this project the introduction of the Revocation Authority and the Inspector, which are relatively new to the Trust Management literature. However, such an ambitious architecture would pose many legal challenges. Hence, A preliminary legal analysis of the ABC4Trust architecture is provided by Camenisch et al. [5] as the basis for a more thorough analysis in the future.

The architecture developers designed it to carefully tackle some of the main Security and Privacy issues. For example, to prevent Credential Pooling where Users would share their issued Credential to access resources they are not allowed to, it is proposed by Camenisch et al. to use Credential binding to a User's secret and, then, the issuer may insist that password or secret should be used to create all the different Credential that would make that secret so valuable that a User would hardly be willing to share. As an extra protection, the binding could be a person-to-device that is distributed in a way that makes it 1-1 relation (smart card for example) [5]. Another example is the possibility to make certain Data inspectable by only certain Legal or pre-agreed-upon Authorities. In this case, the User should generate his presentation token encrypted by the inspector's public key. This feature could be useful in other situations like

treating a bank as an inspector opening encrypted bank account number with payment to verifier as inspecting grounds.

As can be observed so far, the ABC4Trust architecture is innovative in the way it tackles the current Security issues without sacrificing the User's Privacy or the Legal Authorities rights of inspection in case of suspicion. Nevertheless, there are still some Security issues that were not tackled in this proposal. For example, if collusion between the verifier and the inspector happens, no guarantees would be available for the User. An improved approach would be to enforce User notification before de-anonymising any Credential. Moreover, while this architecture provides well-tailored protection for Users Credentials, it offers no protection for the exchanged Data. Although the User might be anonymised, his Data might be linkable to his real identity, depending on the nature of that Data, which would raise some linkability issues.

3.1.6 UMA

The User-Managed Access standard, or simply UMA, is an effort of the UMA workgroup which is part of the Kanatra initiative. Per the project website, the UMA workgroup is concerned with developing standards that enables the web Users to control their Data sharing and service access with interoperable implementations of the standards [59]. A full implementation of the protocol in Java that adheres to the software design core principals like modularity and unit testing is described by Machulak et al. [81].

The UMA developers think that the XAML based approach to Data management are either inflexible or lack the User-centricity mechanisms [6]. Their proposed alternative approach is to implement a protocol named User Managed Access (UMA) which tries to fulfil the following ten requirements for any User managed system:

- Access relationship service
- User-driven policies and terms
- User-managed access relationships

- Auditing
- Requester-Host direct access
- Multiple hosting services
- Entity separation
- Resource orientation
- Representation agnostic access control
- Preservation of Users' Privacy

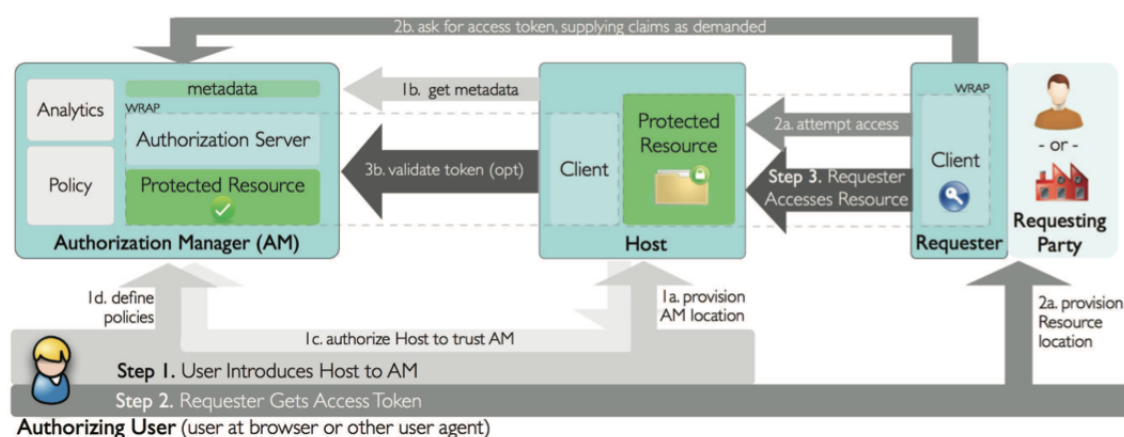


FIGURE 3.6: High-Level Overview of the UMA Protocol [6]

Figure 3.6 shows a high-level overview of the UMA protocol. In that Figure, the authorising User is the User who stores his Data at different hosts while depending on the authorisation manager to deal with Data access requests to ensure that only those requesters who promise to obey the access control policy can get the Data. The Data access restrictions are heavily based on the OAuth 2.0 and, hence, it shall be assumed that all OAuth 2.0 Security features could be implemented in UMA including TLS secure communication [56]. From the protocol specifications [6], it can be seen that it is possible for a User to require in the access policy that a requester should provide certain claims before gaining access. Such claims could be, but does not have to be, signed by a Trusted third-party.

This protocol is one of the latest additions to the Trust Management literature with the advantages of being formally standardised and sponsored as well. Nevertheless, one of its main shortcomings is the fact that it lacks after release protection. There are not even attached Sticky Policies with the released Data in case a genuine requester wishes to adhere to Data access policies set by its original owner.

3.1.7 OAuth 2.0

The OAuth 2.0 Framework is published as an IETF standard Track since October 2012 [56]. Among the main adopters of the OAuth 2.0 Framework is Facebook [82]. Per Hardt, the traditional way for a third-party application to access a resource on behalf of a client, or its owner, is to explicitly ask for the client's login Credentials, typically username/password [56]. Such an approach is inherited with many downsides including:

- Insecure storage of the sensitive Credentials, in plaintext, by some applications.
- Applications get too much access to all the protected resources by the given password.
- Revoking granted access for an application means changing the password that is used by all the other applications.
- Compromising the application means compromising all the Users' Data.

Given the abovementioned issues, the objective of the OAuth 2.0 Framework is to enable applications to get a limited access to online resources on behalf of client. OAuth 2.0 approach is made possible by introducing an authorisation layer which grant authorisation tokens of limited scopes to the third-party applications. The OAuth 2.0 Framework is designed to be used over the HTTP protocol only and it depends on implementing its Security features such as secure messaging, signatures, and encryption on the TLS/SSL protocols to increase the Framework flexibility and not to re-invent the wheel.

In terms of the protocol usability, OAuth 2.0 does not support or mandates a specific form of Security, authentication, or contract negotiation giving the Security engineers full flexibility when tailoring down specific solutions for their systems. Nevertheless, the authorisation layer provided by this protocol should be compatible with the other Security layers in the deployed applications. OAuth 2.0 standard and threat model can be found online in [56]. Plus, some draft implementations are available on the OAuth 2.0 official website [83].

Despite being an IETF standard that is gaining more adoption among the biggest players in the Internet, there are many issues that a system designer should consider before relying on OAuth 2.0. One of those issues is the fact that there is no mechanism to protect against colluding parties like the resource server and application to compromise the resource owner Data.

3.1.8 OpenID Connect

The OpenID protocol is a Single-Sign-On, SSO, protocol that enables individuals to sign in to multiple websites using a single identity managed and provided by a single OpenID host, OP. Among the goals of the OpenID protocol are [84]:

- Accelerating the sign-up process.
- Eliminating the need to manage many accounts/passwords (this could be viewed as a disadvantage because losing one password means losing all the accounts. Exactly the same case as using the email address as the username for every site with the same password everywhere because our brains cannot remember many passwords).
- Giving the User greater control over her online ID with the choice to use the same ID in many websites so that she can take her reputation to other websites she visits (there is still no technical means to transfer the reputation in some kind of universally agreed-upon measures).

- Minimising the Security risks of losing a password because the passwords are not shared with other websites (a claim that cannot be technically proved), the providers of OpenID are more kin to protect Users' Security than others (another claim that cannot be proved), and if a compromise occurs, you can simply change your password (which is the same case with other providers).

The newly introduced OpenID Connect protocol is simply a realization of the OpenID identity authentication on top of the OAuth 2.0 authorization layer [8]. While the original OpenID 2.0 implements its own encryption and message formats to securely communicate the authentication process, a cumbersome and error-prone task for many developers, the OpenID Connect is relying on the OAuth 2.0 layer to accomplish this task by utilising the widely deployed and accepted TLS protocol. At the moment, the OpenID Connect Protocol is a final IETF Standard with various implementations on a plethora of programming languages provided and adopted by many professional organizations such as Google, Facebook, and Microsoft [85]. Despite being a new protocol, it is estimated that there are over half a billion Users' accounts that are ready to be used by OpenID Connect thanks to the major identity providers, such as Google and Microsoft, adopting it [86].

In Figure 3.7, the main components of the OpenID Connect protocol are illustrated on top of the OAuth 2.0 protocol, see Subsection 3.1.7 for more details. The Core component defines the main authentication process and flows based on OAuth 2.0 while the optional Discovery and Dynamic Client Registration components define how clients could identify a new OpenID provider and register with them automatically. The optional Session Management and Form Post Response Mode components aid the client at handling how to manage the returned parameters and responses from the OAuth 2.0 protocol [7].

Figure 3.8, illustrates the abstract authentication flow that is executed by the OpenID Connect Protocol. In this Figure, the End-User wishes to get a service from a Service Provider that is called a Relying Party, RP. To allow the End-User to access the service provided by the RP, the RP forwards the End-User to the webpage of its OpenID Connect identity provider, OP, to authenticate itself. Once the End-User

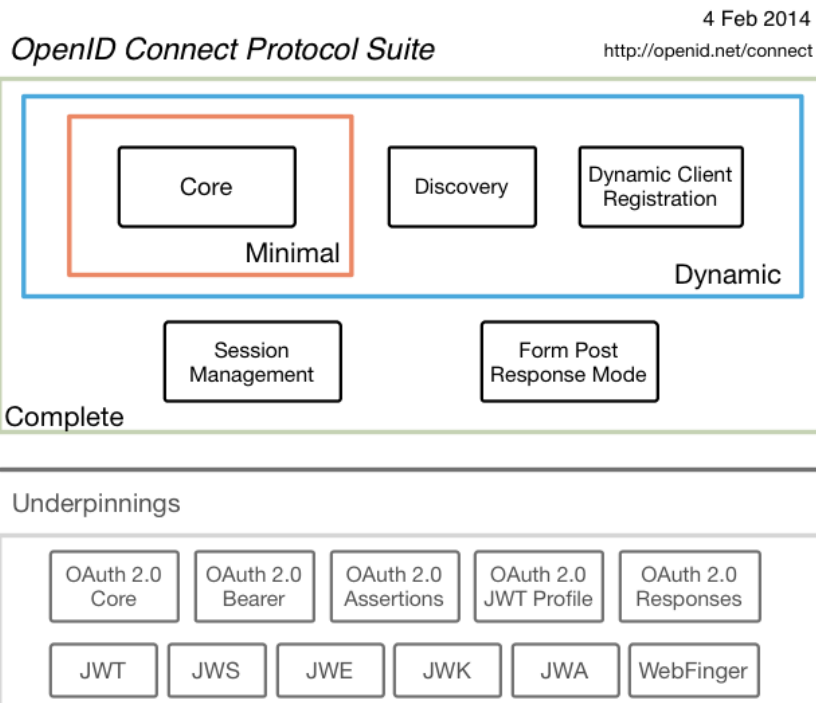


FIGURE 3.7: A General Protocol Suite of the OpenID Connect [7]

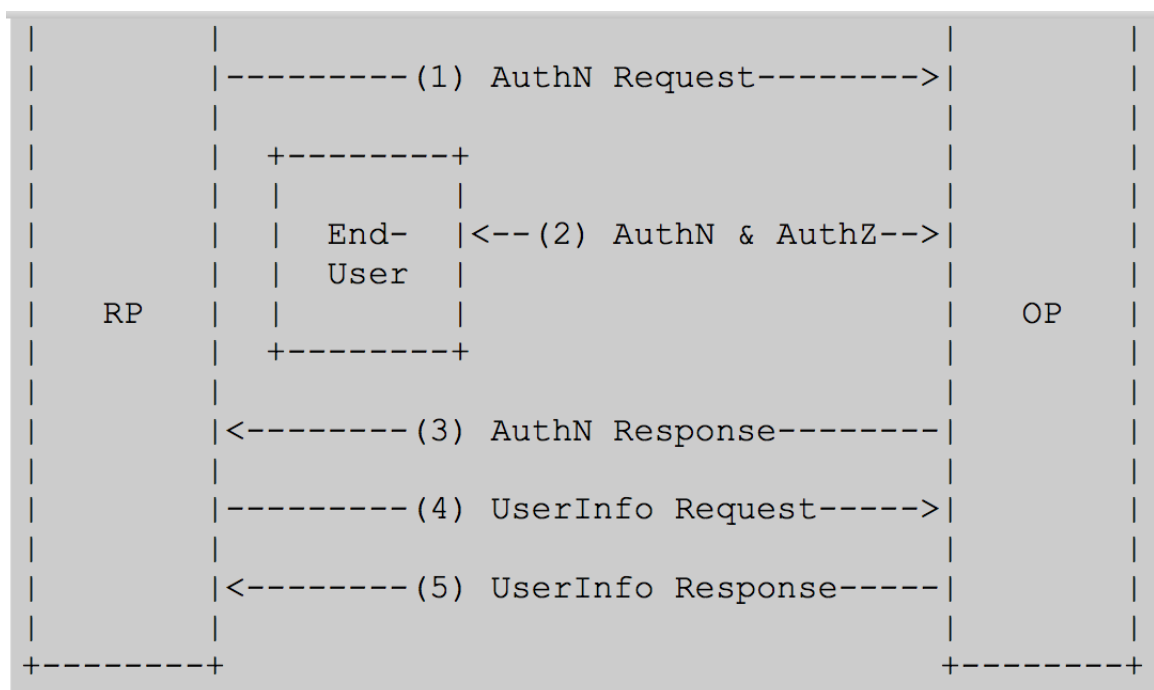


FIGURE 3.8: An Abstract Authentication Flow of OpenID Connect Protocol [8]

authenticates itself, possibly by using a pair of username/password, it would authorize the RP to access a set of its Credentials, such as username and date of birth. The

Op would then send an id token, and possibly an access token as well, to the RP who could then use it to ask the OP for the End-User's Credentials whenever needed [8].

Despite the novel solutions provided by the OpenID Connect protocol, it does have some Security and Privacy loopholes that needs addressing. Per Mainka et al., the automation of the OpenID host discovery and the subsequent client registration introduced in this product opens the door for a set of second-order vulnerabilities. Such vulnerabilities include, for example, malicious clients tricking innocent Users to authenticate themselves through genuine OpenID providers and, then, hijack the returned access tokens allowing the attackers to gain confidential Users' information [87]. The Security issues are not limited to the protocol's design itself but also extends to the implementations as well. Li and Mitchell studied 103 OpenID Connect clients that support the Google's implementation of the protocol and revealed a set of critical Security issues. These issues are caused by deficiencies in Google's implementation as well as the way how the clients are configured to communicate with Google and consume its returned tokens [86]. What we also note is the fact that the OP is always assumed to be Trusted and, hence, there are no rules to enforce encrypting the End-Users' Credentials while stored in the OP servers as well as rules facilitating how such encrypted Credentials would be decrypted by the RP once it is authorized to access them.

3.1.9 Shibboleth

Shibboleth is a federation Single Sign On, or simply SSO, project that enables inter-organisations authentication [58]. This Framework is widely deployed, mainly by higher education institutions due to the support of the JISC advanced institution. Further, it is completely open source released under the Apache Software License [58].

Basically, Shibboleth works by forming a federation of institutions that Trust each other. Then, the participating parties would agree to accept identity tokens from each other proving the identity of their Users without necessarily revealing their whole set

of attributes. Then, when a User of one institution tries to access a restricted resource by another institution that is part of the federation, Shibboleth would basically ask the User to select her ID provider from a given list. Then, Shibboleth enables transferring the required User's attributes from the ID provider to the Service Provider with the option to set attributes exchange policies. Finally, instead of updating the communication metaData among all the participant parties in the federation, the federation would contain a metaData central file that would be updated periodically by each participant [58].

Technically speaking, per Cantor et al., all the messages exchanges are done via encrypted SSL/TLS channels. Moreover, it is recommended for identity providers to sign their messages to mitigate the risk of rouge ID providers' attacks. In addition, the target parameter in the authentication request should be anonymised, by having a static value and, then, tracking the real target by means of state value for example, so that the ID provider do not get extra information about the activities of the ID owner. Finally, if the Users' Privacy is to be considered, it is recommended that a transient ID should be used and that ID should not be used twice to prevent linkability [88].

The apparent problem of Shibboleth is in the fact that it is institutional-centric rather than User-centric [89]. That is because the maintenance of the User authority is not lifelong; once a User is not affiliated with an organisation anymore, he cannot be authenticated to control his generated Data [89]. Moreover, all the involved parties, Service Providers and ID providers, should engage in pre-negotiations to create federations that support the SSO feature which makes it hard for small institution or individual Service Providers or un-affiliated Users with formal organisations to utilize this solution [89]. Another shortcoming of Shibboleth is the lack of Data access rights management after the owner release his Data to the Service Provider. In addition, there are no guarantees against colluding parties among a federation that leaks their Users Data to each other. The worst-case scenario for a User is to get affiliated with a whole corrupted federation where there would be no technical guarantees that his Privacy would be ever respected.

3.1.10 Kerberos

Per Neuman and Tso, Kerberos is a distributed authentication protocol that enables a client to identify its identity to a verifier without exposing confidential Data for the Network eavesdroppers, by means of cryptography [55]. Data integrity and confidentiality are optional features of Kerberos protocol. Kerberos website states that Kerberos is "the most widely deployed system for authentication and authorization in modern computer Networks [90]."

The adopted authentication mechanism by Kerberos is the cryptographic authentication approach in contrast to the inconvenient password approach and the less secure assertion by a Trusted third-party approach [55]. Moreover, Kerberos work by having the Users registering a password, User key, in the authentication servers', or simply AS, DB by means of physically going and registering or possibly by means of asymmetric encryption. Then, if the User wishes to access a resource in a service server, or simply SS, it should send a request to the authentication service, a Trusted party by all the nodes in the system, who would issue a timestamped Ticket-Granting-Ticket, or simply TGT. Note here that the time stamp is to prevent an interceptor from breaking it, a lengthy process, and then impersonating the original client. The timestamp would then be hashed by a common secret between the AS and the SS to prevent the client from tampering with it and, hence, preserving the ticket integrity. If the same client wishes to access another SS, he can just send his request to the AS along with his TGT to get a new session key without the need to use the login Credentials again, in accordance with the Single-Sign-On principle. Once the verification between the client and the server is done, a secure communication channel can be established using the session key, to initiate symmetric cryptography channel, to exchange Data in an integral and confidential manner. However, any software that wishes to use Kerberos need to be Kerborised, which is basically some modifications and upgrades to make it understand and use the Kerberos protocol.

Per Neuman and Tso [55], the main drawbacks of Kerberos V5, the standard Kerberos which was initially developed in 1989, are:

- Ineffective against password guessing attacks, in case Users choose poor passwords.
- A Trusted path to enter the password is required, in other words, the password should not be entered to a malicious software by the User.
- Kerberos is not standalone, it should be integrated with other parts of the system, like the OS for example.
- Kerberos does not offer protection for the entire communication between two nodes in the Networks, it only protects Data exchanges between software that have been modified to incorporate Kerberos.
- Kerberos is not an authorisation software but it can be used to transmit authorisation Data, in other words, a building block for authorisation systems.

3.2 Evaluating the Current Trust Management Frameworks

In this Section, we present extensive tabular comparison of the picked sample Frameworks in Section 3.1 based on how they fulfil the Trust Requirements presented in Section 2.5 as well as the Trust Measures they incorporate from the Trust Measures Spectrum of Section 2.8.

3.2.1 *The Trust Enforcement Properties and Attributes Supported by Current Trust Frameworks*

In this Subsection, we compare the evaluated Trust Frameworks in terms of how they provide the Properties of the Trust Enforcement Requirement that we present in Section 2.6 as well as in Figure 2.2. The results of our comparisons are tabulated in Table 3.1. When it comes to the **Trust Security Property**, the leftmost group of columns in Table 3.1, we notice that all the Trust Frameworks provide some sort of Secure Communication Channels given its vital role to transfer sensitive Data among the Trust Stakeholders. Further, all the evaluated Trust Frameworks also provide some

sort of an Authentication Layer although the complexity of this layer would vary from basic access control like in the case of Kerberos to very advanced Authentication like in case of OpenID Connect. In addition, most of the evaluated Trust Frameworks are flexible enough to accommodate some of the currently available Security solutions and plugins. In fact, many of the Frameworks manuals ask the implementers to choose of the shelf solutions for certain modules of their architectures instead of providing a ready to deploy implementation, see Section 3.1. The exception here are the more specific and mature Trust Frameworks: Shibboleth, Kerberos, and PrimeLife. Finally, all the evaluated Trust Frameworks are open source, which would facilitate independent code check for integrity assurance.

When it comes to the **Trust Privacy Property**, the centre group of columns in Table 3.1, we compare the Anonymity and Pseudoanonymity Privacy Attributes in one column due to their strong relation and the fact that most of the evaluated Trust Frameworks provide some sort of both Trust Attributes. We compare the Continuous Data Control Attribute in two columns: Data Control Prior Release and Data Control After Release. The reason for creating these two Sub Attributes is the fact that most of the evaluated Trust Frameworks provide good measures to control the Trust Stakeholders Data prior to releasing them to other parties but they fail to enforce any control after release which make the Continuous Data Control Attribute unfulfilled. The conceptual solutions provided by TCG and UCON are the only Trust Frameworks that have some sort of Data Control After Release. The problem with TCG is the immaturity of the TSS layer where it still can't attest or capture a lot of relevant Data. More research is needed to get a fully reliable Trust Framework that can provide Continuous Data Control based on the TCG roots of Trust, see Subsection 3.1.1. In contrast, the current direction in realising the UCON conceptual model to provide Continuous Data Control is to utilize TCG solutions with all its inherited issues, see Subsection 3.1.2.

When it comes to the **Legal Authorities Trust Property**, the rightmost group of columns in Table 3.1, we notice that most of the evaluated Trust Frameworks support the Active Authorities Access Attribute in theory. In practice, it is up to the Trust

Frameworks' managers to decide whether to offer this Trust Attribute as well as the Legal Authorities themselves to decide whether to get active and utilize this Trust Attribute. In addition, we see that most of the evaluated Trust Frameworks cannot fully offer the Data Handling Laws Attribute or the Digital Rights Managements Attribute since the lack of the Data Control After Release Attribute. Even the TCG and UCON Trust Frameworks which offer some sort of these Trust Attributes are of limited reliability as we discuss in Subsection 3.2.2. Of course, the legislator should understand the current state of the art where implementing this Trust Attribute is not technically possible yet in most of the scenarios. Finally, most of the evaluated Trust Frameworks appear to be fair competitors in the market except for the Shibboleth because it favours institutions over individuals who are not unaffiliated with such institutions.

Framework / Protocol	Security Attributes				Privacy Attributes			Legal Authority Attributes			
	Secure Communication	Authentication	Compatibility with Security Tools	Open Source	Anonymity & Pseudonymity	Data Control Prior Release	Data Control After Release	Active Legal Authorities	Data Handling Laws	Fair Competition	DRM
TCG	Yes TNC Access policies for groups. Classic authentication & TCG Security tokens	Yes HW/SW multi-tier authentication	Yes Open source. Supports IDS, VMs, Trusted HW, Secure OS, & SW attestation. Provides building blocks based on other technologies	Yes	Yes	Yes HW/SW multi-tier authentication to decrypt Data	Partially Auto encryption & remote attestation	Possible By contract conditions	Possible Full Data control by auto encryption & attestation	Fair Possible unfair competition in absence of proper laws	Possible Full Data control by auto encryption & attestation
UCON	Possible Many prototypes use encrypted containers	Partially Has an Authorization component	Yes Accommodates existing Security solutions	Possible No standardisation	Possible Accommodates anonymity algorithms	Possible HW/SW attestation like in the case of MBA	Partially HW/SW attestation like in the case of MBA	Possible By contract conditions	Possible By proper Enforcement	Fair Possible unfair competition without laws	Yes HW/SW attestation & watermarking

Framework / Protocol	Security Attributes				Privacy Attributes			Legal Authority Attributes			
	Secure Communication	Authentication	Compatibility with Security Tools	Open Source	Anonymity & Pseudonymity	Data Control Prior Release	Data Control After Release	Active Legal Authorities	Data Handling Laws	Fair Competition	DRM
TAS3	Yes Using encryption controlled PEP & PDP	Yes	Yes Accommodates existing Security solutions	Yes	Yes	Yes By Sticky Policies	Partially Auditing engine to rank Trustworthiness	Possible By contract conditions	Partially Design principles based on laws but no after release protection	Good Open source. Could be built over & co-exist with other solutions	No
PrimeLife - PPL	Yes Using encryption	Yes	Yes An access control layer independent of lower layers	Yes	Yes	Yes Restricted access to authorised nodes	No	Possible By contract conditions	Partially Data minimisation & generic Privacy legal principles compliance but no after release protection	Good Open source & no tying-up with a service provider could be inferred	No

Framework / Protocol	Security Attributes				Privacy Attributes			Legal Authority Attributes			
	Secure Communication	Authentication	Compatibility with Security Tools	Open Source	Anonymity & Pseudonymity	Data Control Prior Release	Data Control After Release	Active Legal Authorities	Data Handling Laws	Fair Competition	DRM
ABC4Trust (PrimeLife Sequel)	Yes Using encryption	Yes	Yes A high-level ID MGT independent of lower layers	Yes	Yes	Partially Encryption, revocation for IDs but not Data raising linkability risk	Partially Possible to revoke usage of Credentials. But no control over consumed Data	Possible By issuers allowing Authorities to decrypt & de-anonymise Credentials	Partially Supports EU Data minimisation Act plus utilities to enforce stricter future law	Good Open source & could be built over & co-exist with other technologies	No
UMA	Yes Handling protected resources on different servers	Yes	Yes Focused on Web 2.0	No	Possible Tailoring access for providers to partial profiles	Yes Using access control policies	No	Partially Access policies may conform with laws	Possible By contract conditions but, no after release protection	Good Most implementations open source	No

	Security Attributes				Privacy Attributes			Legal Authority Attributes			
Framework / Protocol	Secure Communication	Authentication	Compatibility with Security Tools	Open Source	Anonymity & Pseudonymity	Data Control Prior Release	Data Control After Release	Active Legal Authorities	Data Handling Laws	Fair Competition	DRM
OAuth 2.0	Yes TLS secure messaging	Partially An Authentication layer to support Authentication systems	Yes Accommodates different ways of authenticating and proving IDs	Yes	Partially Anonymity if providers don't need ID. But no protection against colluding parties	Partially Authorised content release. But, no guarantees against colluding parties	Partially Authorisation server can revoke access but no guarantees against colluding parties	Possible By contract conditions	No No minimal processing, guarantees against colluding parties, or after release protection	Good Open source	No
OpenID Connect	Yes TLS secure messaging	Yes	Yes Accommodates different ways of authenticating and proving IDs	Yes	Partially Anonymity if providers don't need ID but no protection against colluding parties	Partially No Data transfer without User's consent but, no guarantees against colluding parties	No	Possible By contract conditions	No No minimal processing, guarantees against colluding parties, or after release protection	Good Open source	No

Framework / Protocol	Security Attributes				Privacy Attributes			Legal Authority Attributes			
	Secure Communication	Authentication	Compatibility with Security Tools	Open Source	Anonymity & Pseudonymity	Data Control Prior Release	Data Control After Release	Active Legal Authorities	Data Handling Laws	Fair Competition	DRM
Shibboleth	Yes SSL/TLS channels	Yes	No	Yes	Yes Offers Transient identity if Privacy required	Partially Only allowed access without colluding federated parties protection	No	Possible By contract conditions protection	Partially Supports EU principle of necessary processing by proper policies but no after release protection	Limited Favours institutions over ordinary Users who are not affiliated with them	No
Kerberos	Yes Symmetric authentication using Trusted third-party	Partially Basic access control	No	Yes	No	Partially Secure channels & basic access control	No	No Can't support laws by its own	No Can't support laws by its own	Good Open source	No

TABLE 3.1: Comparing the Trust Enforcement Attributes of Figure 2.2 Supported by Current Trust Frameworks

3.2.2 The Trust Flexibility Properties and Attributes Supported by Current Trust Frameworks

In this Subsection, we compare the evaluated Trust Frameworks in terms of how they provide the Properties of the Trust Flexibility Requirement that we present in Section 2.7 as well as in Figure 2.2. The results of our comparisons are tabulated in Table 3.2. When it comes to the **Trust Practicality Property**, the leftmost group of columns in Table 3.2, we notice that most of the evaluated Trust Frameworks offer a fair amount of the Documentation and Support Attribute to help potential developers and deployers getting start. Moreover, most of the evaluated Trust Frameworks are active projects except for the TAS3, PrimeLife, and ABC4Trust academic projects which have already terminated. The same could be said about the Deployment Attribute: all the evaluated Trust Frameworks are already deployed except for the three academic projects which have only some sort of pilot projects that have been partially deployed for experiments. Finally, most of the evaluated Trust Frameworks have serious reliability issues. The exceptions were the more mature and specialised projects that focus on a partial set of the Digital Trust issue. Those Frameworks are OAuth 2.0, OpenID Connect, Shibboleth, and Kerberos.

When it comes to the **Trust Customizability Property**, the rightmost group of columns in Table 3.2, we notice that the Data Classification Attribute is left to the Users to establish themselves by setting suitable access policies for their Data. Further, most of the evaluated Trust Frameworks are neutral and could offer some sort of the Negotiation Attribute as external layers integrated on top of them. Finally, most of the evaluated Trust Frameworks offered some sort of the Delegation Attribute with the exception of the UCON and Kerberos Trust Frameworks.

Project / Framework	Practicality Attributes				Customizability Attributes		
	Documentation & Support	Active Development?	Deployed?	Reliability	Data Classification	Negotiation	Delegation
TCG	Good Available Guides & ready to use applications	Yes	Partially TPM, TSS, TNC, and MTM specifications	Limited Issues with attestation. Plus, basing roots of Trust on the condition of OS integrity isn't enough	Yes TNC aid tailoring access policies for different groups	No	Possible TNC & TSS could aid Authentication layer with delegation
UCON	Limited No official umbrella - scattered research in many papers	N/A No official research umbrella	Partially Many prototypes in the literature	Limited No real-life implementation that could be analysed or a standardisation umbrella	Possible By setting appropriate policies	Yes The main model & some prototypes support negotiation	No UCON is about protecting access to Data not about Identity MGT
TAS3	Limited Various guidelines are provided but, the documents are hard to follow since many parts are incomplete	No Finished on 2011	Partially ZXID engine as a reference implementation of the core TAS3 Security	Limited Still at prototype level without any sort of support	Possible Using Sticky Policies	Yes	Yes
PrimeLife - PPL	Good Full book summarising project results	No Finished on 2011	Partially Some prototypes available as Open Source	Limited Still at prototype level without any sort of support	Yes Using PPL access policies	Possible Policy engine that could accommodate negotiation modules	Yes
ABC4Trust (PrimeLife Sequel)	Fair Good documentation along with video tutorials. But, no reference implementations	No Finished on 2014	Partially Pilot prototypes	Limited Still at prototype level without any sort of support	Yes By setting appropriate access policies	Possible Issuers set conditions & providers set policies for external Negotiation modules	Yes

Project / Framework	Practicality Attributes				Customizability Attributes		
	Documentation & Support	Active Development?	Deployed?	Reliability	Data Classification	Negotiation	Delegation
UMA	Good Easy to access documentation and FAQ	Yes	Yes Three implementations: SMARTAM, Fraunhofer AISEC, and UMA to TAS3	Fair Still in beta but the SMARTAM is up to date implementation of the standard and the Cloud Identity Ltd. provides UMA support	Yes By setting appropriate access policies	No	Yes
OAuth 2.0	Fair Specifications & threat model but, no de-facto implementation or guidelines yet	Yes	Yes By large corporations like Facebook	Good Widely deployed & supported protocol	Yes By setting appropriate access policies	Possible Possible to enforce pre-negotiated terms	Yes
OpenID Connect	Good Plugins for popular environments plus libraries of sample code and specifications	Yes	Yes Over 500M enabled Users' accounts	Good Widely deployed & supported protocol	Yes By choosing what personal attributes could be shared with service providers	No	Yes
Shibboleth	Good Documentations for both developers & deployers	Yes	Yes Mainly by high education providers	Good Research efforts are not deployed until getting stable	Yes By setting appropriate access policies	Possible By setting attributes exchange policies	Yes
Kerberos	Good Documentations tailored for different interest groups	Yes	Yes	Good Widely deployed and stable	No	No	No

TABLE 3.2: Comparing the Trust Flexibility Attributes of Figure 2.2 Supported by Current Trust Frameworks

3.2.3 Current Trust Frameworks on the Trust Measures Spectrum

In this Subsection, we compare the sample Trust Frameworks we picked in Section 3.1 in terms of the Trust Measures Levels they deploy out of the Trust Measures Spectrum we describe in Section 2.8 as well as in Figure 2.3. The results of our comparisons are tabulated in Table 3.3. When it comes to the weakest Trust Measure Level, Cues and Clues of the second column in Table 3.3, most of the evaluated Trust Frameworks got good ranking. That is due to the support those projects are getting from the big players in the market like Microsoft, Intel, and Facebook to name a few. Moreover, most of the evaluated Trust Frameworks utilize some sort of the Policies and Contract Trust Measure, as shown in the third column of Table 3.3. Regarding the Standardisation and Certification Trust Measure, of the fourth column of Table 3.3, all of the evaluated Trust Frameworks have some form of standardisation except for the three academic projects. In addition, it is noted that the Reputation Management Trust Measure, of the fifth column of Table 3.3, is underutilized by the evaluated Trust Frameworks. When it comes to the Access Control Trust Measure, of the sixth column of Table 3.3, all of the evaluated Trust Frameworks supported some sort of Enforcement methods. In regards to establishing Trust by the Identity Management Trust Measure, as shown in the seventh column of Table 3.3, most of the evaluated Trust Frameworks utilize some sort of it or, at least, serves as building blocks for such systems. Finally, most of the evaluated Trust Frameworks failed at utilising the Hard Verification Trust Measure, as shown in the last column of Table 3.3. That is, a minority of the evaluated Trust Frameworks support Continuous Data Control but with limited reliability as we discuss in Subsection 3.2.2.

Project / Framework	Cues & Clues	Policies & Contracts	Standardisation & Certification	Reputation System	Access Control	Identity Management	Hard Verification
TCG	Yes	Yes Uses HW verification, remote attestation & Network level policies	Yes ISO standard	No	Yes Uses HW verification, remote attestation, TNC & Network level policies	Yes	Yes Uses HW verification, remote attestation, & Network level policies
UCON	No No standardisation umbrella	Yes By means of Sticky Policies	No	Possible The MBA realisation ranks Trustworthiness of Users based on behaviours	Yes Access policies enforced using reference monitors	Partially Utilities to protect Users' attributes but isn't mainly concerned with managing Users' Data	Possible The MBA realisation depends on HW roots of Trust like TPM attestation
TAS3	Yes	Yes By means of sticky policies and a Legal Trust Framework	No	Yes Reputation Trust engine to rank nodes based on audit trail	Yes By means of Sticky Policies Enforcement before release	Yes	No
PrimeLife - PPL	Yes	Yes Using PrimeLife Policy Language - PPL	No	No	Yes Using PPL engine to enforce access control rules	Possible A building block for Privacy-preserving access control system	No

Project / Framework	Cues & Clues	Policies & Contracts	Standardisation & Certification	Reputation System	Access Control	Identity Management	Hard Verification
ABC4Trust (PrimeLife Sequel)	Yes	Yes Both client and Service provider can set their own access policies	No	No	Yes Using access policies, encryption, and revocation authorities	Yes Privacy-preserving ID MGT layer over other authentication Frameworks	Yes - Creds. only By access policies, encryption, and revocation authorities
UMA	Yes	Yes Users specified access control policies	Yes IETF Standard	Possible Possible to require a reputation certificate	Yes By deploying OAuth 2.0 to control access	Yes	No
OAuth 2.0	Yes	Partially Only pre-negotiated contracts	Yes IETF Standard	No	Yes By authorisation tokens with variable Security measures	Partially Relies on IDPs to authenticate Users	No
OpenID Connect	Yes	No	Yes IETF Standard	Possible Service Providers could require User's claims coming from Trusted IDP and/or reputation certificates	Yes Deploys authentication layer on top of OAuth 2.0	Yes Users can authenticate themselves by using one ID from a Trusted IDP	Partially - Creds. only Tamper-resistant proofs but no guarantees against malicious IDPs

Project / Framework	Cues & Clues	Policies & Contracts	Standardisation & Certification	Reputation System	Access Control	Identity Management	Hard Verification
Shibboleth	Yes	Yes Attributes exchange policies	Yes Based on SAML standard	No	Yes SSL/TLS to enforce SAML policies	Yes	No
Kerberos	Yes	Partially Trivial access policies	Yes IETF proposed standard	No	Yes Trivial access policies	Possible External plugins could be coded	Yes Symmetric authentication using Trusted party

TABLE 3.3: Themes Comparison of the Evaluated Trust Frameworks

3.2.4 Other Relevant Projects and Frameworks

Since the Trust term in computer science is very broad, it is unrealistic to comprehensively cover all the relevant projects and Frameworks to it. This Subsection lists some relevant projects and Frameworks for reference of any interested researcher. These projects are organised in four different categories: industry/standard bodies led projects, academic projects, deployed authentication and authorisation solutions, and governmental initiatives. Each project or Framework is tagged by the resources it protects and the fields of computing it covers.

Unevaluated Industry / Standard Bodies Led Projects			
Project / Framework	Brief Description	Protected Resources	Covered Fields
TCG	The Trusted Computing Group, TCG, aims to develop open standards to tackle the computing Trust issues by, mainly, rooting all the Trust chains to a Trusted hardware chip, called TPM, that has many built-in Security features like storing sensitive Data and generating secret keys [64]	Generic	Generic
TCG: MTM	The Mobile Trusted Module, MTM, is the sibling of the TPM for the mobile industry [91]	ID	Mobile
TCG: TNC	The Trusted Network Connect protocol, TNC, is an access management standard for Networks [92]	Generic	Client/Server

Unevaluated Industry / Standard Bodies Led Projects			
Project / Framework	Brief Description	Protected Resources	Covered Fields
Common Criteria	An international ISO standard that sets common Security evaluation measures to generate different levels of Security certificates [71]. Such certificate would, in turn, give some Trust to the evaluated product.	Generic	Generic
Central Authentication Service (CAS)	A Single Sign-On protocol that was originated in Yale University 2004 [93]	ID	Client/Server
Higgins 2.0	A cloud-based Framework to protect personal Data sets [94]	Generic	Cloud
OASIS	A non-profit standardisation body to develop open standards to handle online information in many areas including web services and cloud computing [49]	Generic	Generic
OASIS: WS-Trust	A protocol to establish Trust relationships by issuing, asserting, and managing Credentials exchange on top of a secure communication channel established by the WS-security protocol [95]	ID	WS
OASIS: SAML	A protocol to exchange authentication and authorisation information between client and identity providers which help in establishing SSO [96]	ID	Generic
OASIS: XACML	An access control policy language that defines how to enforce authorisation rules using XML syntax [97]	Generic	Generic

TABLE 3.4: Other Industry/Standard Bodies Led Projects

Unevaluated Academic Projects			
Project / Framework	Brief Description	Protected Resources	Covered Fields
PrimeLife	A EU funded project that lasted from March 2008 until October 2011 [98]. The goal of the PrimeLife project is to correspond to the new Privacy challenges that are posed by the emerging web of online social communities, mashup applications, and lifelong storage with nearly unlimited storage capacity [57]	Generic	WS; Client/Server; Social Networks
PrimeLife: Clique	a Privacy enhanced social Network where Users define who has the right to see every single entry [99]	Data	Client/Server; Social Networks

Unevaluated Academic Projects			
Project / Framework	Brief Description	Protected Resources	Covered Fields
PrimeLife: Scramble!	A tool providing audience segregation by encryption by means of implementing a Firefox extension that decrypt any piece of published Data if and only if the publisher has given permission to the other end to read the entry [99]	Data	Client/Server; Social Networks
PrimeLife: Personal Data MOD	An extension to the phpBB forum that reminds the User of his actions that are visible to other audience groups (registered, moderators, owner, everyone) [99]	Data	Client/Server; Social Networks
PrimeLife: Privacy Enhancing Selective Access Control for Forums	A tool that enables a User to set more fine-grained access control policies for his individual entries [99]	Data	Client/Server; Social Networks
PrimeLife: Duddle - Privacy-enhanced Web 2.0 Event Scheduling	A tool that enables social Network Users to create anonymous polls [99]	Data	Client/Server; Social Networks
PrimeLife: The Privacy Dashboard	A Firefox extension that shows to the User what are some of the Service Providers do with his Data (like lasting cookies, third-party content, flash cookies, usage of p3p policies,...). Plus, it enables the User to restrict giving/receiving Data for certain websites or content that utilize some of the predefined practices [99]	Data	Client/Server; Social Networks
PrimeLife: Over-Encrypt	A tool that stores encrypted Data in untrusted Databases while showing it to a selected group of Users via the use of client-side Firefox extensions [99]	Data	Client/Server
PrimeLife: Pri-Views	A Data fragmentation tool that stores most of the Data in an outsourced untrusted server while retaining the small fragments that contains the values or the part that makes the large set meaning full in the owners computer. This has the advantage of not using encryption which slows down the progress and makes it more expensive [99]	Data	Client/Server
PrimeLife: Identity Mixer Crypto Library	A Java library that enables a User to create Credentials and prove ownership of them to enable deploying applications that offer pseudonymity [99]	ID	Client/Server; Social Networks

Unevaluated Academic Projects			
Project / Framework	Brief Description	Protected Resources	Covered Fields
PrimeLife: PET-Uses	A usability questionnaire to enable flexible measurement of a given Privacy Framework in terms of its general usability and how the software aids the Users' understanding and management of Privacy aspects [100]	Generic	Generic
PICOS	A EU funded project to develop technologies that enhance Privacy for mobile communities which lasted from 2008 until 2011 [101]	Generic	Mobile
SSEDIC	A EU funded project to create a thematic Networks for the European eID and is scheduled to last from 2010 until 2013 [102]	ID	Generic
ICT-Endorse	A EU funded project, which lasted from 2010 until 2013, to prepare a legal technical framework for data privacy management. One of the main goals of the project is to develop an open source toolkit to guarantee that personal Data are being handled in legally compliant manner. The other goal is to generate a certification methodology to better evaluate the Trustworthiness of ICT products with respect to Privacy and Data protection [103]	Generic	Generic

TABLE 3.5: Other Academic Projects

Unevaluated Deployed Authentication and Authorisation Solutions			
Project / Framework	Brief Description	Protected Resources	Covered Fields
CoSign: Secure, Intra-Institutional Web Authentication	An open source project originally designed to provide the University of Michigan with a secure single sign-on web authentication system. cosign is part of the National Science Foundation Middleware Initiative (NMI) [104]	ID	Generic
Stanford WebAuth	WebAuth is an authentication system for web pages and web applications [105]	ID	Generic
Facebook Platform	A complete platform that offers ID login, graph access, and other FB functionalities for mobile and web developers [106]	Generic	Generic
Flickr Authentication API	Authentication API using Flickr [107]	ID	Generic

Unevaluated Deployed Authentication and Authorisation Solutions			
Project / Framework	Brief Description	Protected Resources	Covered Fields
Google Accounts Authentication and Authorization	Authentication and authorisation API using OAuth 2.0 to access Google accounts [108]	Generic	Generic
Authentication and Authorization with Yahoo	A set of APIs to authenticate using OpenID, authorise using OAuth 2.0, or doing both operations at once using a hybrid protocol. Single sign on is also provided using BBAuth [109]	Generic	Generic
U-Prove	An unlikable and Privacy preserving partial ID code by Microsoft [110]	ID	Client/Server
OpenAM	An authorisation and authentication management Framework that establishes SSO [111]	ID	Client/Server

TABLE 3.6: Other Deployed Authentication and Authorisation Solutions

Unevaluated Governmental Initiatives			
Project / Framework	Brief Description	Protected Resources	Covered Fields
FICAM	Federal Identity, Credential, and Access Management: A guide for the US official agencies to manage clients' Credentials [112]	ID	Generic
Future ID	A project to create an identity management infrastructure that is scalable, flexible, and preserve Privacy for Europe by combining the existing eID technology with other federated identity management and other Trust Management solutions [113]	ID	Generic

TABLE 3.7: Other Governmental Initiatives

3.3 Strengths of the Current Trust Management Frameworks

In this Section, we highlight the strengths and advantages of the evaluated Trust Frameworks of Section 3.2. This is important so that new Trust Frameworks could try to utilize the well designed and implemented software components, or even concepts, offered by those evaluated Trust Frameworks instead of reinventing the wheel,

specially that most components of the evaluated Trust Frameworks are open-source. Particularly speaking, the evaluated Trust Frameworks did a great job in implementing the Trust Security Property as well as most of the Trust Privacy Attributes with fair enough addressing of the Trust Practicality Attributes.

3.3.1 Strong emphasis on Security Attributes

Given that proper system-security acts as the gate-keeper for any Trust Framework, addressing the continuously arising Security issues is well handled by many computing researchers, specially those designing and developing Trust Frameworks, see Subsection 2.6.1. For that, the evaluated Trust Frameworks are doing an excellent job at implementing the Trust Security Property. That is, most of them offer some sort of secure communication channels and basic authentication. Even if new Security issues emerge, the evaluated Trust Frameworks are flexible enough to be compatible with external Security tools, many of them already utilize of the shelve components such us TLS secure communication protocol. In fact, most of the software components of the evaluated Trust Frameworks are open source which makes it easier to accommodate cutting-edge solutions for future Security issues.

3.3.2 Good realisation of the Privacy Attributes - except for the After Release Data Protection

Given the increased concern for Privacy, this vital Trust Property is getting more attention from computing researchers, see Subsection 2.6.2. This attention can be observed by noting that most of the evaluated Trust Frameworks offer some sort of User Anonymity or Pseudoanonymity, as well as some sort of Data control prior to the Data release. In particular, the Trust Frameworks that are concerned about Identity Management (TAS3, PrimeLife, ABC4Trust, OpenID Connect, and UMA) are the best on these aspects. The Trust Frameworks that offer an Identity Provider, IDP, are designed to support the Anonymity and Pseudoanonymity Privacy Attributes as well as to make it possible for the Data owners to control who gets what of their

Data. Nevertheless, supporting the vital Data Control After Release Trust Attribute is under-researched as we highlight in Subsection 3.4.1.

3.3.3 Reasonable Practicality Attributes

As expected, despite the wealth of novel and cutting-edge solutions found in the evaluated academic projects (TAS3, PrimeLife, and ABC4Trust), they do not offer a good realisation of the Trust Practicality Property. That is expected due to the limited resources they have. In fact, their task is to give inspiration for new and creative ideas to be adopted by the big industrial players and the standardisation bodies.

On the other hand, the standardised Trust Frameworks are offering reasonable realisation of the Trust Practicality Property. That is, they offer fair documentation and support, they are still undergoing active development, and have some sort of deployed prototypes. Yet, they are not all equal. TCG, despite being sponsored and developed by the big players in the industry, is still of limited reliability due to its immaturity. UMA, OpenID Connect, and OAuth 2.0 Trust Framework are thriving in real-life and integrating well to offer a futuristic Trust Framework, as we mention in Section 2.8. Furthermore, Shibboleth and Kerberos Trust Frameworks are well-established and adopted by millions of Users to facilitate federated authentication and basic access control.

3.4 Weaknesses of the Current Trust Management Frameworks

In this Section, we highlight the weaknesses of the evaluated Trust Frameworks of Section 3.2. This is important so that new Trust Frameworks should focus on addressing these shortcomings with novel solutions. The evaluated Trust Frameworks are severely lacking support for the vital Data Control After Release Attribute which, in turn, makes it impossible to offer satisfactory realisation of the Legal Authority Trust Property. Furthermore, while most of the evaluated Trust Frameworks are

flexible in theory, there is a lack of solid realisations of the Customizability Trust Property.

3.4.1 *Lack of Continuous Data Control*

As we mention in Subsection 2.6.2, the Continuous Data Control Privacy Attribute is about giving the Data owners the power to control who access their own Data and how they would handle it after they gain access to it. While many of the evaluated Trust Frameworks come up with solutions to control who could gain access to Users' Data, there is little done to control how such Data would be handled after its release to other parties. For that, we divide the Continuous Data Control Attribute to two separate sub-attributes in our tabular comparisons of Subsection 3.2.1, to give credit to the Trust Frameworks that have tried to offer the Data Control Prior to Release while failed to offer significant Data Control After Release.

Even the Trust Frameworks that offered some sort of Data Control After Release are not of satisfactory level yet. That is, while the TCG Framework offers promising hardware utilities for attestation and remote rules Enforcement, there is a lack of a concrete software layer that translates the theoretical powers of attestation to practical software solutions. In addition, there are various bugs and issues found in the TCG Framework hardware components, as we mention in Subsection 3.1.1, making solutions based on the TCG attestation of limited reliability.

When it comes to the UCON Framework, there is the fundamental problem of lacking a standardisation umbrella, which means that every team could have their very own interpretation of what a UCON System should do and provide. In fact, the main focus of the UCON Framework is to offer solid DRM for creative Data owners, more so than tackling the issue of Data Control After Release. In addition, many UCON implementations rely on the attestation powers of TCG to offer the DRM Attribute which make it inherit the above-mentioned TCG limitations. However, we note that some other novel UCON Data Access Enforcement solutions, such as watermarking,

could be utilized by a Continuous Trust Management Framework to enforce Data Handling Laws after release.

The other researched Trust Frameworks offer less significant implementations of the Data Control After Release Sub-Attribute. The TAS3 Framework includes an Auditing engine to rank the Trustworthiness of the Trust Network Residents, but we could not find any implementation or design documents associated with this engine to verify its potential. The ABC4Trust project offers some revocation of access to Credentials that have been released by Users to Service Providers. That is a good basic Data Control After Release feature, but it is not enough. Data Control After Release is not only about enabling Users to revoke access to their release Credentials, but also about controlling who could access what and for what purpose. Further, this control should not be associated only with Users' Credentials but should also be extended to Users' generated Data. Finally, we draw similar conclusions for the possible access revocation offered by the OAuth 2.0 Framework, in that there are no guarantees about colluding Authorisation Servers that may abuse its access powers.

3.4.2 Lack of solid implementation of the Legal Authorities Attributes

As we mention in Subsection 2.6.3, having an active Legal Authorities Enforcement is crucial to complement any shortcomings of the deployed technical Trust solutions. While some of the evaluated Academic Trust Frameworks focus on the legal aspects of Digital Trust and produced valuable documentations of the relevant laws and how they could be technically satisfied, this vital Trust Enforcement Property is still under-utilized. It is possible in theory to give Legal Authorities access to protected Data in case that is needed, but this is a cumbersome and a controversial task in practice, see Subsection 2.6.3. Further, the lack of the Data Control After Release Sub-Attribute does not only limit the level of possible compliance with the Data Handling Laws or the level of offered DRM but also means losing the chance to log valuable information regarding who accessed what and when. If such logs are available, they would be valuable for the Legal Authorities to view and analyse in case of a dispute between a Data owner and suspected Data breachers.

3.4.3 Lack of solid implementation of the Customizability Attributes

Despite the fact that it is possible, in theory, to support the Trust Flexibility Customizability Attributes, there is little offered in practice apart from the excellent Delegation Attribute offered by Trust Frameworks like TAS3 and OAuth 2.0. We could not find solid software components that would aid the Users to easily classify their Data based on importance or type. Lacking the Data Classification Attribute along with lacking the Data Control After Release Attribute makes it of less value for the Trust Stakeholders to negotiate for Data Handling Laws. That is probably the cause of lacking solid implementations of negotiation software components apart of the experimental components, offered in the evaluated academic projects.

3.5 Why a Continuous Trust Management Framework and What it Should Provide?

In Section 2.1, we show that the notion of Digital Trust is a *state of mind* where the Trust Stakeholders would engage in transactions with the believe that the other party is Trustworthy and will do its best to deliver the best service per the negotiated terms. For that, any Framework claiming to be a Trust Management Framework should make every possible effort, given the current state of the art of technology, to fulfil the listed main Trust Requirements in Section 2.5.

In Section 3.4, we mention three main weaknesses in the state of the art Trust Frameworks. The most critical limitation is the lack of proper Continuous Data Control. The lack of this control raises the doubt among Data owners about how their Data would be handled after they release it to other parties which would, in turn, reduce the level of the perceived Trust by those Data owners. This issue is the most critical limitation of current Trust Frameworks, also because it is a main cause for the other two mentioned weaknesses: *Lack of solid Implementations of the Legal Authorities Attributes* and *Lack of solid Implementations of the Customizability Attributes*. Therefore, we argue that a Continuous Trust Management Framework is a necessity as the natural next step to grow the current Trust Framework to their full potential.

Once this step is taken, it would be easier to strengthen the other weak points we mention in Section 3.4. To deserve the Continuous prefix, the Trust Management Framework should pay more attention to the Continuous Data Control Attribute, see Subsection 2.6.2, during all phases of its Stakeholders' transactions. That is, before and after one party releases Data to another party.

Given the limited amount of resources dedicated to a PhD thesis, it is not realistic to implement a full, ready to deploy, Continuous Trust Management Framework. Rather, our aim is to design a generic Continuous Trust Management Framework along with implementing a minimal prototype version of it that fulfils the basic Trust Requirements of Section 2.5. As we mention in Section 2.8, all of the UMA, OpenID Connect, and OAuth 2.0 Trust Frameworks could be integrated to offer a future Trust Framework. Such a Trust Framework would rely on OAuth 2.0 to take care of the Authorisation task, which in turn could rely on well-established protocol like TLS to establish secure communication, while relying on OpenID Connect to handle Authentication, and UMA to handle the complexities associated with Identity Management. The combination of these projects offer the best collection of Trust Requirements along with the more immature TCG, the unstandardized UCON Framework, and the overly narrow Shibboleth and Kerberos Frameworks. Building on and utilising the combination of the OAuth 2.0, OpenID Connect, and UMA would be the natural step toward implementing our vision for a Continuous Trust Management Framework.

Our Continuous Trust Framework Design along with our minimal prototype are presented in Chapter 4. The theoretical building blocks for the Auditor component that we introduce in the Continuous Trust Framework Design to provide the Continuous Data Control are presented in Chapter 5 while the simulation of the Auditor's performance is listed in Chapter 7.

Chapter 4. Designing the Continuous Trust Management Framework, Prototype, and Problem Demonstration

In Chapter 3, we illustrated why the current Trust Management Frameworks are not sufficient to satisfy the needs of the Trust Stakeholders and, hence, the need for our proposed Continuous Trust Management Framework. In Section 4.1, we introduce the design of our Continuous Trust Management Framework that tries to fulfil the essential Trust Requirements we identified in Section 2.5. Our design pays special attention to the Continuous Data Control Privacy Attribute, given as the weakest point of the current Trust Frameworks in Subsection 3.4.1.

To prove the practicality of our proposed design, we present a sample Prototype in Section 4.2. This Prototype assumes a simple use-case where the Continuous Trust Management Framework is mainly concerned with protecting its Users' emails against unauthorised sharing, which could be detected through a received spam from an unknown sender. This Prototype implements the basic functionalities of the Continuous Framework such as the GUI, authentication, authorisation, Data handling, and Data logging for policy Enforcement purposes. That is, this Prototype proves the practicality of implementing a secure Continuous Framework to handle Users' Data and log the Data transactions just before releasing it to the Service Providers. For the variety of the Trust Management use-cases that could exist in the real world, we argue in Section 4.3 that our sample use-case could be generalised to cover all the other use-cases relating to Continuous Data Control.

The Data Governance Unit, DGU, is a hypothetical unit providing hardware roots of Trust as we present in Section 4.4. This hypothetical unit would offer true Continuous Data Control if adopted by all the Service Providers in the Network. However, the TCG technologies, which are the best candidates to be the base for our DGU Unit, are

not mature yet to allow the development of the DGU, see Subsection 3.1.1. Further, service providers may not be willing to adopt the DGU straight away without natural resistance to change. For that, we present Auditorial Ranking Algorithms in Chapter 5 to utilize the collected logs by the developed Prototype of this Chapter in order to detect the Malicious-Data Abusers based on technologies that are available at hand. That Chapter will also present Algorithms assuming the presence of the DGU Unit to evaluate whether it would be vital for the proposed Continuous Trust Management Framework.

4.1 Continuous Trust Management Framework Design

In Section 3.1, we evaluated some of the main Trust Frameworks in the literature to get a general understanding of the Trust Challenges these Trust Frameworks try to address. Here, we combine the best practices found in these Trust Frameworks in addition to what we thought necessary and missing to present a generic Continuous Trust Framework Design that fulfils the essential Trust Requirements of Section 2.5 with special attention to the Continuous Data Control Privacy Attribute. In Figure 3.1, the evaluated Trust Frameworks are operating within six main layers. Hence, our Continuous Trust Framework Design consists of components existing within these six layers. In addition, our Continuous Trust Framework Design includes a vertical Toolkit layer containing essential utilities to be used by other components residing in different layers. This Design is illustrated in Figure 4.1 where each yellow box represents a software building block that can consist of smaller coding modules as illustrated on the smaller outer boxes. An exception to this rule would be the Hardware Trust layer, which is a mixture of Trusted hardware units and managing coding blocks. Below we talk about each layer in detail:

- **Hardware Trust:** This layer contains the Hardware Roots of Trust. All the Trust that would be built up in this Design would be compromised if this layer fails. Unlike the rest of the layers, some of the functional components in this layer are assumed to be implemented in hardware chips like the TPM, see Subsection

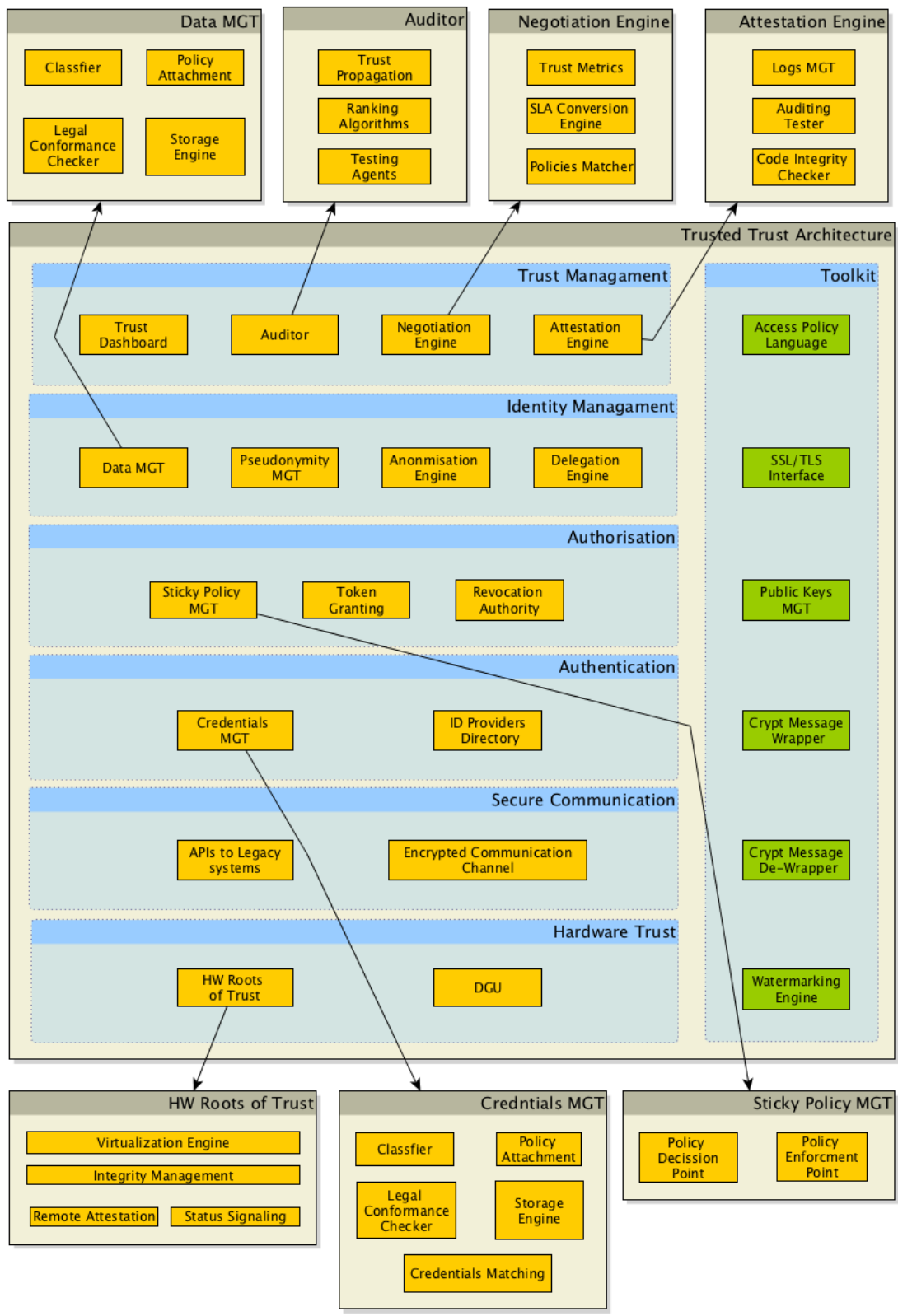


FIGURE 4.1: Continuous Trust Framework Design

3.1.1. The featured components include a Virtualization Engine to facilitate in dividing the stored Data in different environments depending on their Security requirements. Another component is an Integrity Management unit which depends on Remote Attestation and Status Signalling units that are built on the hardware chips. This Integrity Management unit should be able to make sure that the deployed version of the Continuous Framework software is not altered or compromised. An important software building block that operates on top of the hardware roots of Trust is the DGU Unit. This unit would act as a governance unit observing all the Data going in and out the server-side of the Service Provider installing it to prevent any Data handling that is unauthorised by the Data owner. See Section 4.4.

- **Secure Communication:** This is the first Network level layer that is concerned with creating secure communication channels. It contains Encrypted Communication Channels like Kerberos, see Subsection 3.1.10. Further, there should be several APIs to Legacy Systems to communicate with our Continuous Framework.
- **Authentication:** The main authentication building blocks are in this layer. It contains an ID Providers Directory like Shibboleth, see Subsection 3.1.9, as well as a Credentials Matching unit that verifies the identity of the entity trying to Authenticate itself.
- **Authorisation:** This important layer operates after the authentication process success. It is responsible for enforcing the Data associated access rules, Sticky Policies, through the Sticky Policy Management unit. This unit evaluates access requests against the attached Sticky Policies and, if the request is approved, the Token Granting unit would generate a token enabling the requester to obtain the Data through the identity management layer in a similar fashion to how OAuth 2.0 or Kerberos, for example, would work, see Subsections 3.1.7 and 3.1.10. The Revocation Authority unit would enable the Data owner to get full control over her Data or Credentials by revoking a previously granted access

token in a similar manner to how ABC4Trust Framework works, see Subsection 3.1.5.

- **Identity Management:** This layer utilizes the layers underneath it to manage each User profile. It has a Delegation Engine to grant access authorities to other Users or applications in a similar manner to TAS3, see Subsection 3.1.3, or UMA, see Subsection 3.1.6. There are also an anonymization engine, to prevent linking specific Data to specific Users, and a pseudonymity management unit, to help Users to create partial personal profiles that are suitable for certain online contexts. These two units are inspired by the work found in the TAS3 and PrimeLife, see Subsections 3.1.3 and 3.1.4 projects. There is also a Data Management unit that consists of many building blocks. These building blocks include a Policy Attachment unit, a Data Classifier based on the owner preferences, a Data Storage Engine, and a Legal Conformance Checker to check whether a certain policy adheres to certain laws or not. The Storage Unit stores the encrypted Users' Credentials and Data sets. It would not decrypt and release any Credential or Data sets unless for a bearer of a valid access token in a similar manner to the OAuth 2.0 protocol, see Subsection 3.1.7. While most of the previous units appear in other projects like TAS3, see Subsection 3.1.3, PrimeLife, see Subsection 3.1.4, and OpenID Connect, see Subsection 3.1.8, the Legal Conformance unit is not found in the literature but it would be very helpful to satisfy the Legal Authorities, one of the key Continuous Framework Stakeholders.
- **Trust Management:** At this top layer, all the available features in the bottom layers are utilized to provide the Trust Requirements required from a Continuous Framework as listed in Section 2.5. The first component here is the Trust Dashboard which is a presentation layer enabling the human Users to set their access policies, check their current delegations and contracts, check the Trustworthiness rankings of Service Providers prior to engaging in a transaction, revoke previously granted access tokens, and so on. That is similar to the work found in TAS3, see Subsection 3.1.3, and some of the PrimeLife mini-projects,

See Subsection 3.2.4. The Auditor Unit deploys a set of Ranking Algorithms to rank down and ban suspicious Service Providers. To confirm its suspicions, it can create artificial Users, Testing Agents, to interact with the suspicious Service Providers and verify whether they act maliciously or not. The Trust Propagation unit would make sure to propagate the latest rankings to all Users, Service Providers, Legal Authorities, and so on. The Logs Manager keeps interaction records between different Network entities to be used by the Ranking Algorithms in case of disputes. See Chapter 5 for more details about the design of the Auditor. TAS3 and UCON, see Subsection 3.1.2, also rank each entity in the Network based on its behaviour and others' feedback. The Negotiation Engine helps at matching the Users' Trust Requirements with the Service Providers capabilities and policies to facilitate the negotiation process.

- **Toolkit:** This layer is a warehouse of all the necessary and reusable components to facilitate the work in other layers. Unlike the rest of the building blocks in Figure 4.1 that represent required functionalities that could be implemented by combining different software or even hardware components, the building blocks of this layer are solid software solutions that could be utilized by the different building blocks in the rest of the layers. For that, these building blocks are coloured green instead of orange. This layer includes an Access Policy Language similar to XACML or PPL, see Subsection 3.1.4. It also contains an SSL/TLS Interface to deploy the basic Security features found in that protocol instead of rewriting them in a similar manner to OAuth 2.0, see Subsection 3.1.7. There is a Public keys Management unit to facilitate in authenticating and authorising Users. Furthermore, there are Crypting and Watermarking units to help protecting Users' Credentials and Data in accordance with their associated Sticky Policies in a similar fashion to UCON projects, see Subsection 3.1.2.

4.2 Continuous Trust Management Framework, the Simple Prototype

In this Section, we list the details of the simple Continuous Trust Framework Prototype that we developed to prove the practicality and usefulness of our proposed

Design in Section 4.1. We start by listing the technologies we used to develop the Prototype. We then describe the sample use-case we use to demonstrate the problem of offering Continuous Data Control. Further, we discuss the generic design and components of our Prototype, which is built on top of the OpenID Connect protocol. The code of this Prototype is published as an open-source project and can be found on: https://gitlab.com/Continuous_Trust_Prototype/Prototype.

4.2.1 Development Utilities

In developing and executing this Prototype Model, many different software and code libraries were utilized. The main ones are:

- **IDE:** Eclipse Java EE IDE for Web Developers - Indigo Service Release 2
- **Programming Language:** JavaSE 1.6
- **Build Automation Utility:** Maven 3.0.4
- **Database Management:** MySQL 5.1.37
- **Web Server:** Apache Tomcat 7

4.2.2 Problem Demonstrator, the Example Use-Case

In Section 3.1, we saw that the currently available Trust Frameworks already take care of most of the layers and units available in our proposed Continuous Trust Framework Design, see Section 4.1. The main exception is the Auditor Unit, see Figure 4.1, which is vital to offer the Data Control After Release Attribute. There is currently no technical way to verify whether an entity that acquired Data from another would have leaked or Abused the Data it acquired. For that, we present a minimal Continuous Framework Prototype that implements the basic functionalities of the Continuous Trust Framework to serve two main purposes. First, to show the practicality of developing and deploying the Continuous Framework given the currently available

technologies. Second, to show the current problem of lacking the Data Control After Release Attribute, the issue which we tackle by the proposed Auditor's Algorithms of Chapter 5 using the logs that this Prototype is capable of collecting.

The story of our simple use-case is as follows: we assume a Trust Network where Users are interested in getting services done from Service Providers. To authenticate the Users, Service Providers, SPs, would ask for some of the Users' Credentials including their email addresses. Users would normally attach a Sticky Policy, see Subsection 4.4.1, to their Credentials asking not to share them with third-party SPs. Nevertheless, some Malicious SPs, MSPs, would ignore these Sticky Policies and would share these Credentials with third-party MSPs. Those MSPs would utilize the maliciously acquired Credentials to spam the email addresses of their victim Users. At that point, the User would get aware that his Sticky Policies were violated and, hence, he files a complain to the Auditor. Without any Auditor Algorithms, the Auditor would simply reply back to the User saying that he has no idea which is the MSP violated his Sticky Policies.

In Section 2.4, there are many different types of attacks and threats facing the entities' Data and Credentials. To focus our efforts, the **scope of our example use-case** only considers one type of attacks that is widely common: **Basic Active Attacks on Users' Credentials**. That is, Targeted 2.0 Attacks, Passive Attacks, or any type of Attacks against Users' Data sets is out of scope for our Prototype and the following simulation experiments. Further, we only consider attacks from the MSPs against the Users and not the other way around. In this Prototype, we assume genuine Auditor, IDP, and Users despite that they could act maliciously in real-life. Also, we consider one type of User's Credentials Abuse, that is email spamming.

It is true that security toolkits could effectively minimise the threats of the Basic Mass Attacks, as we show in Section 2.4. Nevertheless, we noted in that Section that the current security toolkits are not effective in tackling the extended "malware version" of the Basic Mass Attacks. In our use-case, the Users are tricked into trusting an MSP that should not be trusted and, hence, they do not hesitate to share their personal data with that MSP who would, in turn, spam them back. For that, we consider

the illusion provided by the MSPs as a form of social engineering, or simply malware, that could not be tackled by conventional security toolkits. After all, no current Trust Framework offers real protection against this form of attack as we show in Section 3.5.

In Section 4.3, we argue that our use-case could be generalised to cover a wide range of other Continuous Data Control Use-Cases. Further, in Section 7.1.2 we argue that despite the limitation of our Prototype's scope, the Prototype provides solid grounds to protect against other types of threats that are not considered in this scope.

A summary of the main entities that we include in our Prototype is as follows:

- **User:** this entity represents an end User wishing to exchange her Credentials with a Service Provider to get a service.
- **Service Provider, SP:** this entity represents a Service Provider aggregating Users' Credentials, full name and email address, before providing the service. Some SPs are bad and, hence, leak (sell) the Data to MSPs.
- **Malicious Service Provider, MSP:** this entity represents a malicious SP that gets (buy) Users' Credentials from other SPs. It uses the obtained Data to abuse victim Users with spam messages.
- **Auditor:** this entity represents the Authority in the Continuous Framework. Any spam victim could report it to the Auditor who should find out who is the guilty SP. The Auditor could then access all the stored logs related to the submitted Cases to judge who are the guilty SPs. Each Case includes the details of the reporting User (name, email) as well as a copy of the spam message (sender name, sender email, date, title, and body). The Auditor has the option to add a comment to the Case, change the status of the case from open to close, and choosing a convicted SP from the list of all the SPs in the system.
- **OpenID Connect Identity Provider, IDP:** this entity represents the server that stores Users Credentials and control access to them in addition to maintaining access logs.

- **Simulation Manager:** this entity represents the simulation website where both Users and SPs could register in the Network and start interacting. During the simulation setup, the admin could make certain SPs affiliated with Malicious Service Providers so that they forward to them any Credentials they receive from the Users who interact with them. For simulation purposes only, the admin can also access the logs maintained by the MSPs to check whether the Auditor’s initial judgments are correct or not. That is, it is possible, but not implemented, to create an index showing the percentage of correct judgments made by the Auditor.

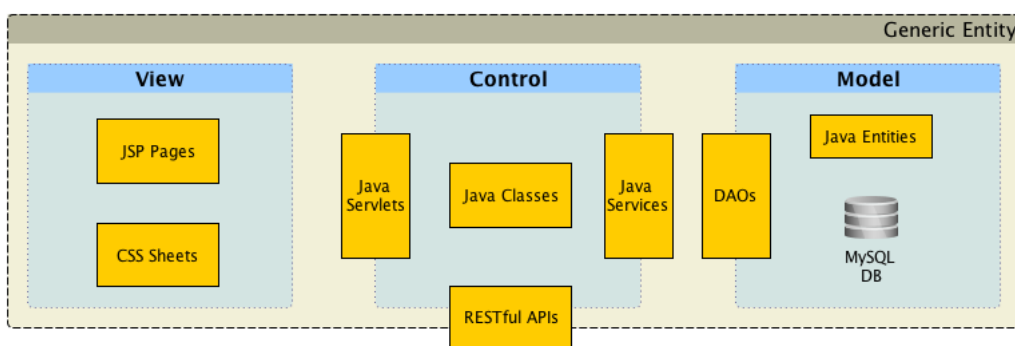


FIGURE 4.2: The MVC Generic Design Pattern

4.2.3 *Prototype Implementation Based on OpenID Connect*

Here we briefly describe our Prototype implementation of the use-case presented in Subsection 4.2.2. In implementing this Prototype, we realised the SPs, MSPs, Auditors, and the Simulation Manager as RESTful Web Services interacting by exchanging JSON tokens, see [114] for more details about RESTful Technology. Because a Network consisting of only one SP, one MSP, and one Auditor is not realistic, we created a number of replicas of each entity by overlaying the original Web Services. That is, while each duplicate of SPs or MSPs would act independently per the simulation settings, they would share exactly the same business logic and the GUI theme. Modifying the original SP or MSP Web Service would result in automatic updates to all the overlays depending on it. All the Web Services are developed using the Spring

MVC Framework to decouple the different layers of the Web Services code that, in turn, improves scalability, usability, and Security, see [115] for more details about this Framework. We list screenshots for the different implemented entities in Appendix A while we show our generic MVC design in Figure 4.2. The components of Figure 4.2 work as following:

- **View:** This is the View, or Presentation, layer consisting of JSP display pages and CSS styling sheets. This layer communicate with the Control layer by means of HTTP Requests sent to the Java Servlets of the Control layer.
- **Control:** This is the Control layer where the business logic is implemented. The first gateway to this layer is the Java Servlets which accepts HTTP Requests coming from the View layer. For simple Data CRUD operations (Create, Read, Update, Delete), the Servlets will directly forward the requests to the Java Services that would, in turn, forward it to the Data Access Objects, DAOs, in the Model layer. If the HTTP Requests are to communicate with another entity, another User or SP for example, then the request would be directly forwarded to the RESTful APIs to send JSON messages to the intended entity. If the entity wish to run some Algorithms, for example an Auditor wish to update its Rankings, then this task would be carried out by the Java Classes.
- **Model:** This is the Model, or Data Storage, layer. The gateway to this layer is the Data Access Objects, DAOs, that performs the requested Data CRUD operations, see [116] for more details about DAO objects. These objects could be extended to work as a filter prohibiting any unauthorised operation, perhaps against the Sticky Policies of the requested Data. The DAO objects utilizes object oriented Java classes to represent each of the entities or Data sets that are stored in the DB. When fetching or persisting the MySQL DB, the retrieved or persisted Data would be realised by the DAO objects as Java Entities giving an extra level of abstraction from the details of low level SQL operations. We achieve this abstraction by using the Java Persistence API, JPA 2, specification, see [117] for more details.

Having described how each entity of our Prototype is implemented, we now describe how they interact within the Prototype Framework. First, this Prototype is working as a minimal Continuous Framework incorporating the OpenID Connect, MITREid Connect implementation [118]. There are many advantages for choosing the MITREid Connect implementation including:

- Implements the authentication and authorization tasks of our Prototype. In fact, it relies on the OAuth 2.0 protocol for authorization and the OpenID Connect protocol for authentication [1].
- A reasonably documented and maintained Java implementation.
- Achieves single sign-on and exchanging Credentials with consents.
- The OAuth2 protocol provides mechanisms to implement revocation authorities and its JSON tokens, which are used to exchange Credentials. That could be extended to attach Sticky Policies.
- Relies on Spring Security layer to provide secure communication Channels.

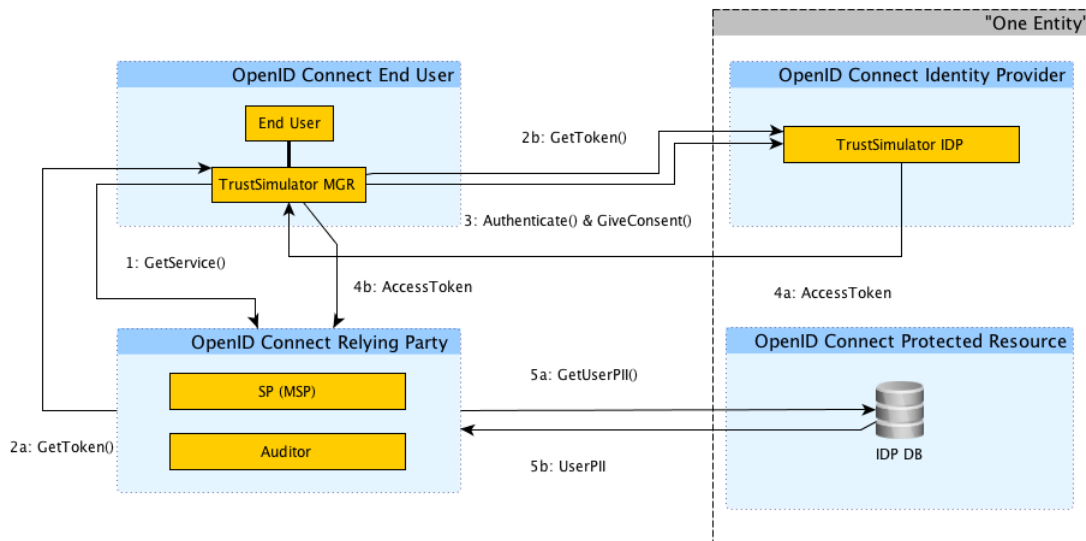


FIGURE 4.3: Authentication Flow using OpenID Connect

Our implementation of the Continuous Framework, powered by the OpenID Connect, includes OpenID Connect providers, IDPs, so that SPs could access the Users'

Credentials using those IDPs per the pre-agreed Sticky Policies even if the Users go offline. In addition, the IDPs would maintain logs of all the Data accesses so that these logs could be submitted to an Auditor upon its request to investigate reported spam Cases. In the OpenID Connect terminology [119], the Users are realised as End Users while the SPs, MSPs, and Auditors as Clients. Figure 4.3 shows how a typical Authentication Flow in our Prototype looks like. The details of this flow are as following:

- **Step 1:** An End User, through the Simulation Manager website, tries to interact with a Client. The Client could be an SP, or an MSP posing as a genuine SP, offering a service which End Users would be interested in. A Client could also be an Auditor, which End Users would like to submit their spam Cases to him.
- **Step 2a:** The Client would reply back to the User request by asking to get an Access Token to present for the IDP to Authenticate the User and acquire some of its Credentials.
- **step 2b:** The User would, in turn, forward the Access Token request to the IDP.
- **Step 3:** The User would authenticate herself to the IDP, using her stored Credentials and password combination for example, and would give consent that she agrees to grant the access token to the requesting Client.
- **Step 4a:** The IDP would generate an Access Token and send it to the User.
- **Step 4b:** The User would, in turn, forward the Access Token to the requesting Client.
- **Step 5a:** The Client would send the Token to the IDP asking to access the User's Credential(s).
- **Step 5b:** The IDP would fetch the Credentials and send it back to the requesting Client and that ends the Authentication and Data release process.

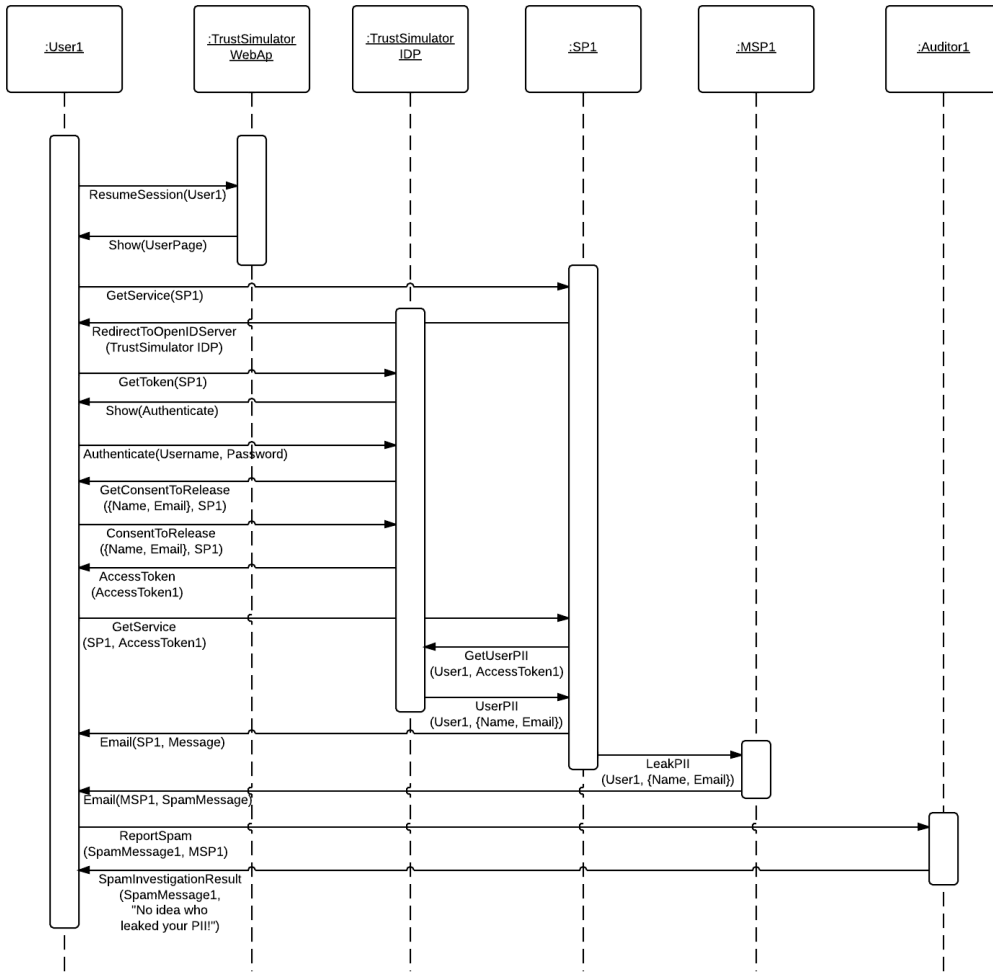


FIGURE 4.4: Malicious Interaction Sequence Diagram

The lack of Continuous Data Control Problem is illustrated in Figure 4.4 that shows a malicious interaction sequence diagram. In that Figure, if the Client the End User releases her Credentials to happened to be an MSP posing as a genuine SP_1 , then SP_1 would forward the Credential to another MSP_2 . MSP_2 would spam the victim User and the User would have no idea who is MSP_2 since she has never dealt with him before. As a result, it would submit a spam report Case to the Auditor. The Auditor would get access to all the logs recorded by the IDP. Nevertheless, it would not be able, in most Cases, to decide who is the guilty SP of leaking the User's Credentials since there are no Ranking Algorithms deployed yet in the Prototype. In Chapter 5, we introduce the theories to establish a useful Auditor that can analyse the recorded logs by the IDP and make some judgements about the guilty SPs based

on them.

4.3 Generalising the Continuous Data Control Problem

So far, we have used a simple email spamming use-case, which we introduce in Subsection 4.2.2, to demonstrate the problem of lacking the Continuous Data Control After Release Attribute. Our initial Prototype collects Data logs about every single Data transaction to aid our proposed Auditor entity in the quest to identify and eliminate the MSPs from the Trust Network, see Chapter 5 for more details about our proposed Auditor.

It is true that the spamming problem where the Auditor is mainly concerned with providing Continuous Data Control for only one Credential, the email-address, is relatively simple. Nevertheless, our proposed Auditor along with the Defensive Strategies we identify in Chapter 6 could be utilized to protect other types of Credentials against Basic Active Attacks, but not necessarily against Passive Targeted Attacks, see Subsection 2.4 for more details about the possible types of attacks. That is, the Auditor would record logs of all the Users' Credentials released in the Network. Further, as long as the MSP is generating a wealth of Abuse Cases, our Auditor should be able to detect its malicious behaviour by analysing the recorded logs using the Algorithms we describe in Chapter 5 and per the rules of the Defensive Strategies of Chapter 6.

While the Abuse in our current sample use-case is noticed by the spam message the victim User receives, there are other signals to look after for different types of Credentials' Abuse. For example, each User, or perhaps the Auditor itself, could deploy a simple software that would utilize the main available search engines to run regular searches for the User's Credentials like the full name, address, and contact info. If these Data appear in a website where it is not supposed to appear, then that would trigger an Abuse that should be reported to the Auditor. Another Abuse trigger would be a User reading his Credentials printed in the newspaper or to get billed for an unknown credit card transaction. In Section 7.1.2, we argue that our proposed Auditor could provide reasonable Continuous Data Control even with a limited number of

reported Abuse Cases, for rarely used Credentials or hard to detect Abuses. Further, we argue in that Section that our proposed Auditor could also protect against Passive Targeted Attacks provided that our DGU Unit gets implemented and widely adopted by SPs.

4.4 DGU, the Hypothetical Data Governance Unit

In our proposed Prototype of Section 4.2, we utilize state of the art technologies like OpenID Connect and OAuth 2.0. In this Section, we present the design of the hypothetical Data Governance Unit, or simply DGU. The DGU is designed to play a vital role in empowering the future Continuous Framework at protecting its entities' Credentials and Data even after releasing them to other parties. The current design assumes the availability of the TCG technologies, TPM and TSS in particular, that are described in Subsection 3.1.1. Given that these technologies are still novice with many challenges to overcome before being widely adopted, we have not implemented the DGU Unit yet. Rather, we present here a use-case that could motivate more research and development in the TCG field.

Our DGU design has few essential requirements to work as expected and eliminate the threat of a malicious DGU. These requirements include:

- **Open Source DGU Implementation:** so that independent programmers could participate in its development while others could evaluate its integrity and Trustworthiness.
- **Server-Side Installation:** of the DGU software by SPs wishing to prove they are Trustworthy.
- **TPM Chip Enabled Servers:** to build on the hardware roots of Trust it provides. That is, the TPM chip would be used to verify that the SP is deploying a **Non-Tampered DGU Software** as it is maintained in the DGU official website.

- **Disabling RAM Access:** to prevent the SP from attempting to sniff the keys utilized by the DGU to encrypt/decrypt the Users' Data or even to sniff those Data while they are processed in the RAM, see [120] for more details about RAM sniffing attacks.

In more details, the SP server Database would have the sensitive Users' Data encrypted, depending on the agreed policy with the User. The DGU would install a DAO gatekeeper that controls access to the Database. That is, it would do the encryption to persist the Data and the decryption to allow the SP to use the Data provided that the SP gives a valid reason according the attached Sticky Policies. In case the reason is valid, the DGU would decrypt the requested Data, handle it back to the SP, and log this action in its record to refer to it in case of dispute or the User reporting Abuse of her Data. In Subsection 4.4.1, we give more details about what are the Sticky Policies and what purposes they could associate with the Data or Credentials they are assigned to. Further, we describe the generic design of the DGU Unit and the main tasks it would perform in Subsection 4.4.2.

The hypothetical unit would be a powerful Trust enabler that, if adapted by all entities, would enforce a very high-level of Trust on everyday transactions. Nevertheless, there are many challenges facing its realisation. As we mention earlier, it relies on stable and mature implementations of the TCG technologies, which is not a reality yet, as we discuss in Subsection 3.1.1. Further, even if it exists, people would be very suspicious to adopt it due to its restrictive powers, see [29]. Even if it got widely adopted by the different entities residing in the Trust Network, we suspect performance issues when it comes to continuous encryption, decryption, and screening operations as we describe them in Subsection 4.4.2. Further studies beyond this thesis are required to determine the exact computational costs of our proposed DGU Unit and how to optimize it.

4.4.1 *DGU Sticky Policies*

Setting up Data access control rules is an active research area with various languages and protocols to achieve, see Section 2.8. The Sticky Policy technique is an advanced way of cryptographically bounding the access control rules to a corresponding piece of Data or Credential as described by the TAS3 developers [3]. As the pioneers of the Sticky Policy technique, TAS3 developers support different methods to integrate it within the Simple Access Protocol, SOAP, that is widely used in the legacy Web Services to exchange XML structured Data. Such integration methods include expanding the Data model corresponding to the exchanged piece of Data to include its Sticky Policy or to wrap the whole XML payload in a custom TAS3 defined layer so that it includes the Sticky Policy [3]. For Sticky Policy to be used in our Prototype of Section 4.2, it should be realised as an extension to the JSON message that is being used in our RESTful implementation rather than an extension to the SOAP message.

When it comes to defining the policy itself, available standards like P3P can provide a wealth of detailed purposes and Users-roles, providing access rights to certain groups of Data, see Subsection Section 2.8. That is, each piece of Data would have a list of acceptable purposes to be offered by a specific list of entities, admins for example, allowing them to handle this piece of Data per their supplied purpose. For the purpose of our generic design, we grouped the possible purposes that could be supplied by the SP to its DGU to decrypt a piece of stored Users' Data as following:

- **View:** this purpose is to say that the SP wishes to view the requested Credential. It is vague why the view is requested and, hence, it would look suspicious. Even if the Sticky Policy permits such a purpose, this SP would be in the shortlist of suspicious SPs if the User reported a compromise on this Credential. Probably, the Auditor would associate this purpose with Targeted Attacks against a particular User rather than a Mass Classic Attack, see Section 2.4.
- **Process:** this purpose is usually about retrieving a collection of Users' Data or Credentials to process them. For example, to get some statistics about the

Users' demography or interests. Because it is not important for the SP to know the identity of each User to execute such processes, the DGU could utilize an Anonmisation Engine to anonymous the returned collection for the SP, see [25].

- **Communicate:** this purpose is about communicating with a User via email, phone, social media, or any other possible way. For such a communication to happen, the DGU should reveal the requested communication medium in a similar fashion to the View purpose. This pose the risk that an MSP may communicate genuinely with a victim User and, then, utilize the revealed communication medium for malicious activities, like spam in our sample use-case. For that, the DGU could deploy a Communication Engine that would take the message from the SP and send it, on its behalf, to the requested User without revealing its contact address to the SP. This analogy also applies to payment requests, credit history validation, telemarketing, and so on.
- **Share with Third-Party:** this purpose is about sharing Users' Data with a third-party SP. Sometimes Users would agree to allow some SPs to share their Data with third-party SPs to send them, for example, relevant offers or perhaps to check their credit history. Nevertheless, the third-party might be an MSP who would Abuse the acquired Data. When an SP who has already *Installed DGU* shares some of its Users Data to other SPs that has not *Installed DGU* yet, the result would be losing the Sticky Policy Enforcement provided by the DGU. We have considered this issue in the Threat Model of our proposed Auditor, see Subsections 5.8.4 and 5.8.7.

4.4.2 *DGU Generic Design*

Figure 4.5 shows the main components required for the DGU to function. Note that this Figure omits the building blocks belonging to the SP and the Auditor that are not directly related to the DGU. The listed Components are as following:

- **DGU DAO:** this Data Access Object is the gatekeeper of the SP Database. All the Credentials that the SP would acquire from the IDP would be encrypted

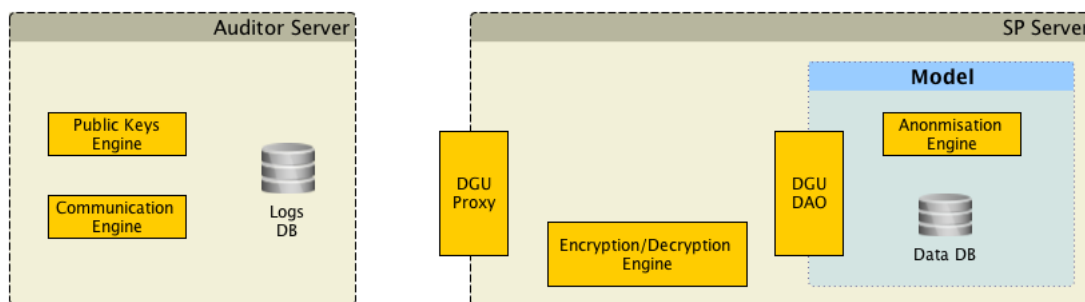


FIGURE 4.5: DGU Generic Architecture

with the public key of the DGU installed in the SP server. That is, only the DGU DAO is able to decrypt the Data that are stored in the SP's own Database. The DGU DAO would only decrypt a piece of Data if the supplied purpose is valid per the requested Data Sticky Policy.

- **Anonymization Engine:** this engine would be used to anonymise a requested piece of Data when the identity of the Data owner is irrelevant to the SP's retrieval request. An example would be the case of a Process request.
- **Encryption/Decryption Engine:** this is a generic unit that could be used by the SP whenever it needs to encrypt anything. That is, the SP is not allowed to encrypt anything unless by using this unit. This is essential to prevent the SP from smuggling Users' Data through encrypted messages sent to other associated MSPs. This unit would screen all encryption requests to figure out whether it contains some of the released Users' Data, through genuine purposes. If such Data is found, it would check whether the granted release authorise sending it to the intended destination or not. If not, this unit would refuse the encryption request.
- **DGU Proxy:** this proxy would screen all the outgoing Data from the SP's server to make sure that no Users' Data are leaving without proper permit from the DGU Unit. If this proxy detects an encrypted output Data that was not signed by the Encryption/Decryption Engine, then it should not allow it to leave the server.

- **Public Keys Engine:** this engine would take care of maintaining the public keys of the Users, SPs, and installed DGUs for the purpose of exchanging the Data with only authorised entities.
- **Communication Engine:** this engine would take care of fulfilling the communication request from the SPs intended for the Users without revealing the communication address of the Users to the communicating SPs.

4.5 Beyond the Continuous Trust Framework Prototype

In Chapters 2 and 3, we have studied the Trust Management Frameworks as whole entities trying to learn their current strengths that we should maintain as well as their weaknesses that we should tackle. In Section 3.4, we argued that the most serious weakness of the current Trust Management Frameworks is the lack of Continuous Data Control Privacy Attribute, see Subsection 2.6.2 for more details about this Attribute. For this reason, we argued for the need for a Continuous Trust Management Framework in Section 3.5 at the end of Chapter 3.

In this Chapter, we proposed a generic Continuous Trust Management Framework Design that is based on the best practices and components found in current Trust Management Frameworks such as OpenID Connect, UMA, OAuth 2.0, and TCG technologies. In addition, we included new components in that design to support the Continuous Data Control Privacy Attribute in order for our proposed Framework to deserve the Continuous prefix. Namely, the two main components that we introduced in that design are: The Auditor and the DGU components. To prove the practicality of our proposed design, we have implemented a sample Prototype of it that contains the basic functionalities, based on a modified version of OpenID Connect protocol. In that Prototype, we have shown clearly the main weakness of the current Trust Management Frameworks that we are trying to tackle: the inability to protect Users' Credentials after they are released to third-parties as well as the inability to detect the Abusers, i.e. the Lack of Continuous Data Control Attribute that we discussed in Subsection 3.4.1.

In Section 4.4, we proposed the Data Governance Unit, DGU, to enforce the Continuous Data Control on the exchanged Data among the Trust Stakeholders. Nevertheless, this DGU Unit is still an abstract concept that is not yet implemented and, hence, cannot provide an immediate solution to the current lack of Continuous Data Control. Further, we envision that even after the successful implementation of the DGU unit, it would be a controversial software that would receive a great deal of resistance from Service Providers to install due to its excessive powers and restrictions it imposes on those who opt to install it, in addition to its high computational cost.

In the following Chapters, we take a different approach to what we were doing so far. Instead of studying the Trust Management Frameworks as whole units, we focus our attention on the newly introduced components of this Chapter, the Auditor and the DGU, to explore how they would be implemented and what effect they would have in a bootstrapping environment where SPs are not very keen on adopting the DGU. That is, we introduce in Chapter 5 the basic Ranking Algorithms, the Basic Deployment Rules, and a set of supporting Utilities to help the Auditor in analysing the collected logs of interaction that our Continuous Trust Management Framework is currently able to collect, see Subsection 4.2.3.

In Chapter 6, we implement a simulation model to model how a large number of Trust Stakeholders would interact and how to optimize the Ranking Algorithms and Deployment Rules of the Auditor to minimise the effects of the Malicious Service Providers, MSPs. By the end of that Chapter, we get to know a set of Defensive Strategies each with its own strengths and weaknesses. In real-life scenarios, smart Auditors should dynamically alter its deployed Defensive Strategy based on its perceived environmental settings. Further future studies should incorporate these Defensive Strategies into the Prototype of this Chapter and study how they would work in the real life out of the Simulation Model

Chapter 5. Trust Auditor Conceptual Design and Algorithms

In Chapter 3, we justify the need for a Continuous Trust Management Framework. That is, we compared the state of the art Trust Frameworks and showed that while they excel in fulfilling most of the fundamental Trust Requirements Properties of Section 2.5 and Figure 2.2, they fail at providing Continuous Data Control as we show in Subsection 3.4.1. In Chapter 4, we proposed a Continuous Trust Framework Design that utilizes the existing features of the state of the art Trust Frameworks as well as introduces new building blocks to support the Continuous Data Control Attribute. Further, we present in that Chapter a prototype of our proposed Continuous Trust Management Framework that implements the basic features of the design as well as aggregates interactions logs and enables spam victims, a special use-case for Data Abuse that could be tackled with Continuous Data Control, to report the Abuse spam they receive to the Auditor Unit of our Continuous Framework. As we show in Section 4.3, the general detection and reporting of the Data Abuse Cases would be a similar job to the implemented special case in our prototype, the email-address Abuse Case.

After reporting the Data Abuse Cases to the Auditor, the similarities between the state of the art Trust Frameworks and our proposed Continuous Trust Management Framework ends. Our Auditor Unit would analyse all the relevant Data Abuse Cases to detect the Malicious Service Provider, MSP, that is guilty for the reported Case. In this Chapter, we introduce a novel set of algorithms, building blocks, and deployment rules that could be utilized by the Auditor in its quest to detect MSPs. In Chapter 6, we simulate the introduced defensive tools of this Chapter to construct a set of Defensive Strategies that are suitable for different environmental situation as well as varying Counter-Attacking Strategies.

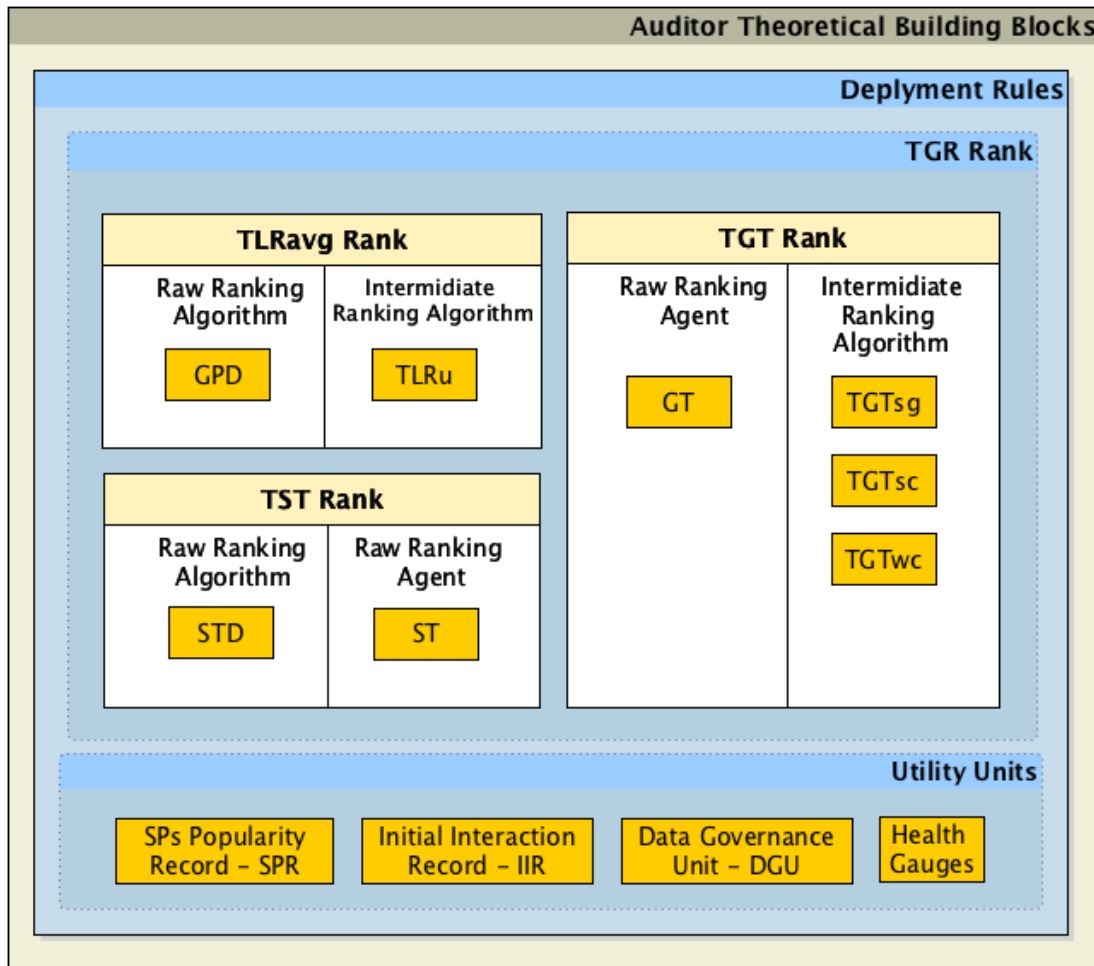


FIGURE 5.1: Auditor Theoretical Building Blocks

5.1 Basic Elements

In this section, the main Ranking Approaches are introduced along with their main components. The Auditor's ranking process assigns a public Rank to each SP in the Network. This rank is called the Global Trust Metric, TGR. As illustrated in Figure 5.1, the TGR Rank is dependent on the results of three different Ranking Approaches:

- **Average of Local Trust Metric, TL_{avg} :** a Ranking Approach that is based on how close an SP usually appears in the Abuse logs just before getting the Abuse. The closer it is, the higher the probability of maliciousness is and, hence, the lower the Rank, see Section 5.2.

- **Sole Testing Trust Metric, *TST*:** a Ranking Approach that is based on Abuse received after dealing with a sole SP indicating that it is almost definitely that this SP is the Abuser, see Section 5.3.
- **Group Testing Trust Metric, *TGT*:** a Ranking Approach that is based on Abuse received after dealing with a group of suspected SPs indicating a possible collation between some of the group MSPs, see Section 5.4.

Each of the above-mentioned Ranking Approaches would consist of internal components that could be categorised as:

- **Raw Ranking Algorithm:** an Algorithm that assigns a specific Rank for each SP based on certain patterns in the received spam cases.
- **Raw Testing Agent:** an artificial User created by the Auditor to testify whether an SP, or a group of SPs, are acting maliciously.
- **Intermediate Ranking Algorithm:** an Algorithm that generates SP Ranks by processing the generated Ranks by Raw Ranking Algorithms and Agents.
- **Final Ranking Algorithm:** an Algorithm that generates SP Ranks by incorporating the generated Ranks by all the Intermediate Ranking Algorithms and Agents. In Figure 5.1, the three internal *TGR* blocks: ***TLR_{avg}* Rank**, **TST Rank**, and **TGT Rank** are all considered to be Final Ranking Algorithms. In turn, the **TGR Rank** itself is also a Final Ranking Algorithm that is based on the results of its internal Final Ranking Algorithms.

Figure 5.1 also shows a set of Utility Units which the Auditor need to calculate and interpret the *TGR* Ranks. These Utility Units could act as **Selection Algorithms** that aid the Auditor in selecting suspicious SPs for the following Raw Testing Agents Algorithms:

- **SPs Popularity Record - SPR:** an Algorithm that sort all the SPs by their popularity based on the frequency of their appearing in the reported Data Abuse Cases, see Subsection 5.6.1.

- **Initial Interactions Record - IIR:** a Record containing all the Initial Interaction Lists, *IILs*, where each list contains the interactions that took place by a given User's Credential prior to receiving the first Abuse Case on that Credential. This list would also include all the Popular SPs, PSPs, that this User have dealt with prior to that Abuse Case using other Credentials for the purpose of detecting Weak Colluding Attacks, see Subsection 5.6.2.
- **Data Governance Unit - DGU:** a hardware Utility Unit to enforce Hard Trust in the Trust Network. If an SP *Installs DGU*, it should be impossible for it to handle an acquired Credential without compliance to its sticky policies. In addition, DGU would record all internal accesses to the Credential as well as all its approved sharing destinations, see Section 4.4 and Subsection 5.6.3.
- **Health Gauges:** a set of powerful sensors to provide feedback reports about the health status of the Trust Network, see Subsection 5.6.4.

In Figure 5.1, it is shown that the Deployment Rules block contains the TGR Rank Final Ranking Algorithm as well as the Utility Units. That is to indicate the fact that the Auditor is able to decide who are the bad MSPs based on the Deployment Rules of its setup, see Section 5.7 where we describe a set of basic Deployment Rules. In Chapter 6, we conduct a series of Simulation Experiments to optimize these basic Deployment Rules. The outcome of those experiments is a set of Defensive Strategies, each with its own strengths and weaknesses.

5.2 TLRavg Ranking Approach

In this Section, we describe the basic building blocks of the TLR_{avg} Ranking Approach and how they would work as they appear in Figure 5.1. In a nutshell, the Gradient Probability Distributor, *GPD* Raw Algorithm evaluates the guilt probabilities for each Service Provider, SP, that appear in all the reported Data Abuse Cases by the victim Users. This evaluation is based on how close, chronologically, an SP appears in the Abuse logs to the timing of the Abuse. The closer it is, the higher the

probability of maliciousness and, hence, the lower the Rank the SP would get. The $TLLR_u$ Intermediate Algorithm then aggregates the SPs Ranks that are local to each victim to construct the Local Trust Metric Record, or simply $TLLR_{U_i}$, that contains Trust Ranks that are local to the U_i . The $TLLR_{avg}$ Final Ranking Algorithm then creates global Rankings for each SP by averaging its local Ranks aggregated form all the $TLLR_u$ Records of all the Network's Users.

5.2.1 Gradient Probability Distributor

This is a Raw Ranking Algorithm for the $TLLR_{avg}$ Ranking Approach, as shown in Figure 5.1. This Algorithm assumes that the last SP who got the User info is more likely to be the Data Abuser while the first SP in the chain is the least suspicious. The reasoning behind this Algorithm is common sense. That is, if the victim was a happy Network User for a fairly long time before getting a Data Abuse just after dealing with a new SP for the first time, then that new SP is probably the MSP behind the Data Abuse. The GPD is listed in Algorithm 5.1, summarised in Equation 5.1 and explained in Example 5.1.

Algorithm 5.1: Gradient Probability Distributor, GPD

- Each reported case, C_i , has n SPs that are presented in chronological order.
- Assign each SP_j in C_i a guilt probability, $GPD_{C_i}(SP_j)$ that initially equals its order j divided by n .
- Divide each $GPD_{C_i}(SP_j)$ by the sum of all the generated guilt probabilities in C_i , that is: $\sum_{k=1}^n \frac{k}{n}$.
- If the same SP_x appears m times in a given C_i , then treat each appearance as an independent SP_j . Finally, $GPD_{C_i}(SP_x) = \sum_{l=1}^m GPD_{C_i}(SP_x)_l$

$$GPD_{C_i}(SP_j) = \frac{j}{n * \sum_{k=1}^n \frac{k}{n}} \text{ or simply: } GPD_{C_i}(SP_j) = \frac{j}{\sum_{k=1}^n k}$$

where n is the number of SPs in C_i and j is the position of the SP

$$\text{And: } GPD_{C_i}(SP_x) = \sum_{l=1}^m GPD_{C_i}(SP_x)_l \quad (5.1)$$

where m is the number of times a distinct SP_x appears in C_i

Equation 5.1: Gradient Probability Distributor, GPD

Example 5.1: GPD Processing

If $C_1 = \{SP_3, SP_3, SP_4\}$ Then:

$$GPD_{C_1}(SP_3) = \frac{1}{3+2+1} + \frac{2}{3+2+1} = \frac{1}{2}$$

$$\text{And } GPD_{C_1}(SP_4) = \frac{3}{3+2+1} = \frac{1}{2}$$

5.2.2 Local Trust Metric Record

This is an Intermediate Ranking Algorithm for the TLR_{avg} Ranking Approach, as shown in Figure 5.1. After evaluating all Cases, C_s , for a given User, U_i , the Auditor can generate a Local Trust Metric Record, or simply TLR_{U_i} , that contains Trust Ranks that are local to the U_i . Each TLR_{U_i} should contain a set of Local Trust values for each SP that the given U_i has dealt with so far. Such value would be referred to as $TL_{U_i}(SP_x)$ for any SP_x that U_i has dealt with. The highest value for $TL_{U_i}(SP_x)$ would be 100, which means that the SP_x is fully Trustworthy in the context of its transactions with U_i . On the other hand, the lowest value would be 0.

The $TL_{U_i}(SP_x)$ value is determined solely by the sum of all the guilt probabilities for SP_x for all the reported Cases by U_i in a negative linear relationship. That means,

the guiltier SP_x within the context of U_i is, the less locally Trustworthy, $TL_{U_i}(SP_x)$. That yields Equation 5.2 which is demonstrated in Example 5.2.

$$TL_{U_i}(SP_x) = \left(1 - \frac{\sum_{j=1}^n GPD_{C_i}(SP_x)}{n}\right) * 100 \quad (5.2)$$

where n = the number of Cases, in the context of U_i , where SP_x appeared

Equation 5.2: Local Trust Metric, TL

Example 5.2: Calculating $TL_U(SP)$

If U_1 has the following Cases:

$$C_1 = \left\{GPD(SP_4) = \frac{2}{3}, GPD(SP_3) = \frac{1}{3}\right\},$$

$$C_2 = \left\{GPD(SP_2) = \frac{1}{3}, GPD(SP_3) = \frac{2}{3}\right\}$$

Then:

$$TL_{U_1}(SP_2) = \left(1 - \frac{\frac{1}{3}}{1}\right) * 100 = \frac{2}{3} * 100 = 66.67\%,$$

$$TL_{U_1}(SP_3) = 1 - \frac{\frac{1}{3} + \frac{2}{3}}{2} = \frac{1}{2} * 100 = 50\%,$$

$$TL_{U_1}(SP_4) = 1 - \frac{\frac{2}{3}}{1} = \frac{1}{3} * 100 = 33.33\%$$

That would give us:

$$TLR_{U_1} = \{TL_{U_1}(SP_2) = 66.67\%, TL_{U_1}(SP_3) = 50\%, TL_{U_1}(SP_4) = 33.33\%$$

5.2.3 Average of Local Trust Metric Record

This is the Final Ranking Algorithm for the TLR_{avg} Ranking Approach, as shown in Figure 5.1. This Algorithm creates a global sense of the local TLR_U s aggregated from all the Network's Users by creating the Average TLR_U s Record, or simply TLR_{avg} . This Record contains the average TL values for each SP within the context of all the different Users who interacted with it as shown in Equation 5.3 and demonstrated in Example 5.3.

$$TL_{avg}(SP_x) = \frac{\sum_{i=1}^n TL_{U_i}(SP_x)}{n} \quad (5.3)$$

where n = the number of Users who interacted with SP_x

Equation 5.3: Average of Local Trust Metric, TL_{avg}

Example 5.3: Calculating $TL_{avg}(SP)$

If the Trust Network has two Users: U_1 and U_2 , and the corresponding $TLLR_{US}$ are presented in Tables 5.1 and 5.2 respectively, then the generated $TLLR_{avg}$ would look like Table 5.3.

SP	TL(SP)
SP_2	83.33%
SP_3	50%
SP_4	60%

TABLE 5.1:
 $TLLR_{U_1}$

SP	TL(SP)
SP_1	50%
SP_3	75%
SP_4	50%

TABLE 5.2:
 $TLLR_{U_2}$

SP	$TL_{avg}(SP)$
SP_1	50%
SP_2	83.33%
SP_3	62.5%
SP_4	55%

TABLE 5.3:
 $TLLR_{avg}$

5.3 TST Ranking Approach

In this Section, we describe the basic building blocks of the TST Ranking Approach and how they would work as they appear in Figure 5.1. In a nutshell, the Sole Testing Distributor, STD , Raw Ranking Algorithm assigns each SP that appear in the log of a reported Data Abuse Case an equal guilt probability. The less SPs appearing in a single Case log, the more the probability that one of them is an MSP. The Sole Tester, ST, Raw Ranking Agent is an artificial User created by the Auditor to deal with a sole suspicious SP. If the ST Agent receives a Data Abuse, this would indicate that the suspicious SP is almost definitely an MSP. The TST Final Ranking Algorithm then is dependent on the aggregated $STD_{C_i}(SP_x)$ evaluations for each SP_x within the

Network whether such evaluations come from the *STD* evaluations of the reported Data Abuse Cases or from ST Raw Testing Agents reports.

5.3.1 Sole Testing Distributor

The Sole Testing Distributor, *STD*, is a Raw Ranking Algorithm for the TST Ranking Approach, as shown in Figure 5.1. In 5.8.6, it is shown that the *GPD* Raw Ranking Algorithm could be easily fooled by Simple Data Abusing Algorithms, like delaying the Abuse so that the abusing MSP would look innocent while an innocent SP might get ranked malicious by the *GPD* Raw Ranking Algorithm. For that reason, this additional Ranking Algorithm is introduced. When the *GPD* Raw Ranking Algorithm processes a reported Abuse Case, it would give worse Ranks to those SPs that the User have dealt with just before getting the Abuse. In contrast, the *STD* processor would give all the Case's SPs an equivalent Rank. This unified ranking function has a negative linear relationship with the number of total SPs appearing in the Case. In other words, the less SPs appearing in the Case, the more likely that one of them is malicious regardless of how soon the interaction took place prior to the Abuse. If an SP_j appeared more than once within a Case C_i , then this Algorithm would count only one appearance and remove all the rest from the calculations. The final *STD* function is shown in Equation 5.4 and explained in Example 5.4. It should be noted that while the *GPD* and the *STD* Raw Ranking Algorithms have similar concepts, they differ in the fact that the generated *GPD* values are guilt probabilities, used to generate $TLLR_{US}$ Ranks, while the generated *STD* values are already SP Ranks. That is, a high *GPD* value indicates a more suspicious SP while a high *STD* value indicates a less suspicious SP.

An interesting scenario would arise if the concerned Credential has a sticky policy permitting sharing it with another SP, see Subsections 4.4.1 and 5.8.4. In that special, but common scenario, it would be more accurate to include all those SPs who acquired that Credential through sharing in the *STD* calculations. That would be only possible in case that SP_x has already *Installed DGU*, see Subsection 5.6.3, since it would enable tracking all the SPs that SP_x has shared with them this Credential. Since SPs who

Install DGU should not be able to directly Abuse Users' Data, the generated list of SPs who acquired the Abused Credential should be filtered by removing all those SPs who have already *Installed DGU*. The filtered list of SPs should be processed by the *STD* Algorithm, which is listed in Algorithm 5.2.

Algorithm 5.2: The Sole Testing Distributor, STD

- When Processing a Case C_i , remove all duplicate SPs from the evaluation list.
- If an SP_x within the evaluation list has already *Installed DGU*, then all those SPs that have acquired the Abused Credential by sharing with SP_x , or SP_y who originally got it by sharing with SP_x , should be added to the SPs evaluation list.
- After extracting all the SPs who acquired the Credential through sharing, all those SPs who *Installed DGU* should be removed from the evaluation list.
- Given n is the final evaluation list size, the *STD* value should be obtained by subtracting $\frac{1}{n}$ from 1 and multiplying the result by 100 to obtain a percentage Rank.

$$STD_{C_i}(SP_x) = (1 - \frac{1}{n}) * 100 \tag{5.4}$$

Where n is the number of SPs in C_i

Equation 5.4: Sole Testing Distributor, STD

Example 5.4: STD Processing

If $C_1 = \{SP_3, SP_3, SP_4, SP_6^{DGU}\}$

And $SP_{6(Sharing-Tree)} = \{SP_2^{DGU}, SP_3\}$

And $SP_{2(Sharing-Tree)} = \{SP_5\}$

Then $C'_1 = \{SP_3, SP_4, SP_5\}$

And $STD_{C'_1}(SP_3) = STD_{C'_1}(SP_4) = STD_{C'_1}(SP_5) = (1 - \frac{1}{3}) * 100 = 66.67\%$

5.3.2 Sole Tester Ranking Agent

In Equation 5.4, there is an interesting special case where in a given C_x , there is only one SP in the log, SP_y . That would yield an STD_{C_1} value of 0 meaning that it is definite, in the perspective of this Algorithm, that SP_y is actually the malicious Data Abuser to blame for C_x . Of course, that is not necessarily true if the User has previous Cases containing an SP_z that has decided to Abuse now, just on time after this User happened to interact with SP_y right after reporting an older Abuse Case. However, if this is the initial interaction for this User using this Abused Credential, see Subsection 5.6.2, and it is not believed that SP_y has shared the acquired Credential with another SP, see Subsections 4.4.1 and 5.8.4, then it could be concluded that SP_y is definitely the Abuser.

The Sole Tester, ST, is a powerful Raw Ranking Agent that is designed based on the above-mentioned special case. Its aim is to verify whether a suspicious SP is really behaving maliciously or not. In this Algorithm, it is assumed that if a Trusted User, an artificial User created by the Auditor, has only dealt with one SP_x in its entire lifetime and it manages to get a Data Abuse, then it is definitely that the Data Abuser is that sole SP_x which this User has dealt with. That leads to the conclusion that SP_x should be ranked malicious with $STD(SP_x) = 0$ Rank. Of course, this assumption might be falsified if the acquired Credential has been shared with a malicious MSP, Subsection 5.8.4. In case the Sticky Policy of the Abused Credential does not allow sharing, then it would be definite that SP_x is acting maliciously and it is highly recommended for the Auditor to ban it from the Trust Network immediately. Even if the Sticky Policy allows sharing, a wise Auditor should consider to ask any detected SP as a potential sole Abuser to either *Install DGU* or get banned from the Network regardless of its current Rank. Such Deployment Rules are discussed in Section 5.7 as well as in Chapter 6 where different Defensive Strategies are optimized.

It should be noted that a smart MSP may decide to use a Dropping Attacking Algorithm where it would decide not to Abuse a certain Credential or a certain User to fool the ST Agent, see Subsection 5.8.6. To counter such attacks, the ST Agent

should maintain continuous interactions with its associated SP and the Auditor should replace that ST Agent after a time threshold. The ST process is listed in Algorithm 5.3.

Algorithm 5.3: The Sole Tester, ST

- Based on a Selection Algorithm, select a suspected SP_x for ST testing.
- Create a new U_x to act as a Sole Tester, $ST(SP_x)$.
- Let $ST(SP_x)$ interact only with the selected SP_x .
- If $ST(SP_x)$ receives a Data Abuse, then obtain the evaluation list, process it by the STD Algorithm, update the $TSTR$ Record, stop verifying SP_x , and kill $ST(SP_x)$.
- Stop ST testing, return $STD(SP_x) = 100\%$, and kill $ST(SP_x)$ after reaching a time threshold $\tau_{ST-maxLife}$ without receiving a Data Abuse.
- If, after waiting for a time threshold $\tau_{ST-idle}$, $ST(SP_x)$ does not get any Data Abuse, then $ST(SP_x)$ should engage in a new transaction with SP_x and repeat the above steps.

5.3.3 Sole Testing Metric Record

This is the Final Ranking Algorithm for the TST Ranking Approach. It is dependent on the aggregated $STD_{C_i}(SP_x)$ evaluations for each SP_x within the Network whether such evaluations come from the STD evaluations of the reported Data Abuse Cases or from ST Raw Testing Agents' reports. The Sole Testing Trust Metric, $TST(SP_x)$, Algorithm for $STD(SP_x)$ Ranks is just like how the $TL_{avg}(SP_x)$ Algorithm is for the $GPD(SP_x)$ guilt probabilities. All the different $TST(SP_x)$ Ranks should be stored in a special Record called the Sole Testing Trust Record, or simply $TSTR$. The entries of this Record can be calculated using the Equation 5.5, see Example 5.5.

$$TST(SP_x) = \sum_{i=1}^n STD_{C_i}(SP_x) \quad (5.5)$$

Where n is the number of Cs where SP_x appears.

Equation 5.5: Sole Testing Trust Metric, TST

Example 5.5: Calculating $TST(SP)$

If the Auditor received three Cases and their corresponding STD s are presented in Tables 5.4, 5.5, and 5.6, then the generated $TSTR$ would look like Table 5.7.

*Note that if the sticky policy of the Abused Credential in $U_2 : C_1$ prevents sharing with a third-party, then SP_5 would be definitely the Data Abuser and, hence, $TST(SP_5) = 0$ despite all its STD values in the rest of the Cases.

$U_1:C_1$	
$SPs :$	SP_1, SP_3, SP_6
$STD_{U_1:C_1} =$	66.67%

TABLE 5.4: $STD_{U_1:C_1}$

$U_1:C_2$	
$SPs :$	SP_2, SP_3, SP_5, SP_6
$STD_{U_1:C_2} =$	75%

TABLE 5.5: $STD_{U_1:C_2}$

$U_2:C_1$	
$SPs :$	SP_5
$STD_{U_2:C_1} =$	0%

TABLE 5.6: $STD_{U_2:C_1}$

SP	TST(SP)
SP_1	66.67%
SP_2	75%
SP_3	70.84%
SP_5	62.5%
SP_6	70.84%

TABLE 5.7: TSTR

5.4 TGT Ranking Approach

In this Section, we describe the basic building blocks of the *TGT* Ranking Approach and how they would work as they appear in Figure 5.1. In a nutshell, the Group Tester Raw Ranking Agent, GT is an artificial User created by the Auditor to deal with a suspicious group, G, of SPs. If the GT Agent receives a Data Abuse, this would indicate that there is at least one MSP among the G members or, if all sole MSPs are caught using the TST Ranking Approach of Section 5.3, the existence of a collusion between some of the G members. The TGT_{sg} , TGT_{sc} , and TGT_{wc} Intermediate Ranking Algorithms assign colluding Rankings to each SP based on different ways of interpreting the logs submitted by both the GT Agents and genuine Users. The TGT_{sg} Ranking Algorithm considers only the GT Agents reported Cases and counts how many times the same reported G appears in the rest of the reported Cases to generate the colluding Ranks. The TGT_{sc} also considers the GT Agents reported Cases and counts how many times subgroups of G appears in the the rest of the reported Cases to generate the colluding Ranks. The TGT_{wc} considers reported Cases coming from both the GT Agents and Users and counts how many times subgroups of G appears in the rest of the reported Cases to generate the colluding Ranks. The interesting thing about the TGT_{wc} Algorithm is the fact that it adds SPs that have dealt with the victim User but not necessarily with the Abused Credential of the evaluated G. The TGT Final Ranking Algorithm then is dependent on a biased weighted average of the ranks of the three Intermediate Algorithms.

5.4.1 Group Tester Ranking Agent

While the ST Agents Algorithm is so powerful that it could give a definite prove that a particular SP is acting maliciously, it will not work very well in case of colluding SPs. That is because such SPs, especially popular ones, would try to imply whether a new User is in fact an artificial Testing Agent or not, see Subsection 5.8.7. To uncover such collations, the Group Testing Ranking Approach, *TGT*, is introduced. At its core lies the Group Tester Raw Ranking Agent, GT. Basically, the GT Agent

is another artificial Agent, just like the ST Agent, see Subsection 5.3.2. Instead of interacting with only one suspicious SP like how the ST Agent works, the GT Agent would interact continuously with a group of SPs, G , to give them confidence that he is a genuine User rather than an artificial one. When the GT Agent receives a Data Abuse, it would be definite that at least one of those SPs is malicious or have shared the GT Agent's Credential with an MSP, see Subsection 5.8.4.

The GT Agent methodology is simple. It would always be assumed that there is an ST Agent for each SP within G and all those ST Agents yield negative results, i.e. no malicious acts detected. Hence, the GT Agent would imply that there are at least two colluding SPs within G . The smaller G is the stronger the probability that its members are colluding. Let's call the group of tested SPs at a given moment G' where, $G' \in G$. Once a Data Abuse is received during testing, G' would be added to a special Record called the Malicious Groups Record, *MGR*. Of course, if SP_x has *Installed DGU* and shared the GT's Credential with SP_y , then SP_y should be included in G' . The *MGR* Record is useful for all of the *TGT* Intermediate Ranking Algorithms.

The trivial assumption that there is an ST Agent for every SP being tested by the GT Agent could be either forced or relaxed by the Auditor, depending on its Deployment Rules, see: Section 5.7 and Section 7.3. Even if it is relaxed, i.e. the assumption is not always true, the results of the GT Agents' testing would still be good educated

guesses. The GT Agents' mechanism is listed in Algorithm 5.4.

Algorithm 5.4: The Group Tester, GT

- Based on a Selection Algorithm, the Auditor should create a new U_x to act as a new GT_x to test a group of suspicious SPs, G_x , that contains n SPs.
- GT_x would randomly select an $SP_x \in G_x$ for the next interaction.
- When GT_x gets a Data Abuse, update the MGR by adding G'_x , which is the group of all the SPs that GT_x has interacted with so far, i.e $G'_x \in G_x$ as well as all the SPs that got GT_x Credential through sharing by $SPs \in G'$. Stop verifying G_i , and kill GT_x .
- Stop repeating and kill GT_x after reaching a time threshold $\tau_{GT-maxLife}$.
- If, after waiting for a time threshold $\tau_{GT-idle}$, GT_x does not get an Abuse, it would repeat the above steps.

5.4.2 Simple Colluding Group Metric

This is an Intermediate Ranking Algorithm that is dependent on the MGR entries. As stated in 5.4.1, the smaller the colluding G is, the stronger the probability that its members are colluding. On this ground, the Simple Colluding Group Trust Metric Record, $TGTR_{sg}(G)$ is created by copying the G_i s from the MGR Record and assigning each of which a $TGT_{sg}(G_i)$ Rank based on Equation 5.6.

$$TGT_{sg}(G_i) = 1 - \frac{2}{n} : \text{where } n = \text{number of SPs } \in G'_i \tag{5.6}$$

Equation 5.6: Group Trust Metric Rank for Gs, $TGT_{sg}(G)$

Since it is possible that a specific SP_x appears in n Gs. Hence, this Ranking Algorithm would assign SP_x a ranking value that is equivalent to the Rank of the smallest G

it appears in per Equation 5.7. The individual $TGT_{sg}(SP)$ Ranks are stored in the TGT Record for SPs, $TGTR_{sg}(SP)$.

$$TGT_{sg}(SP_x) = MIN(TGT_{sg}(G), SP_x) \quad (5.7)$$

: $\forall TGT_{sg}(G) \in TGTR_{sg}(G)$ where $SP_x \in G$

Equation 5.7: Group Trust Metric for SPs, $TGT_{sg}(SP)$

5.4.3 Strong Colluding Group Metric

This is another Intermediate Ranking Algorithm for the TGT Ranking Approach. This Ranking Algorithm is also dependent on the MGR entries and tries to overcome some of the weaknesses of the TGT_{sg} Algorithm, see Subsection 5.4.2. As stated in Subsection 5.4.1, the smaller the colluding G is, the stronger the probability that its members are colluding. While the TGT_{sg} Ranking Algorithm is a good attempt in the quest to discover malicious colluding entities, it suffers from the fact that once the collation size gets reasonably large, the TGT_{sg} Ranks would always be high and, hence, would increase the collation's members Trustworthiness rather than reducing it. It is true that it is normally difficult and expensive for an attacking entity to sustain a large collation. Nevertheless, it is quite possible for a GT, especially when launching an Abuse Delay Attack of Subsection 5.8.6, to end up with a very large $TGT_{sg}(G)$ entry. That is because by the time it manages to get a Data Abuse, its testing Credential would have been shared with many innocent SPs as well as other MSPs who are not part of the suspected collation. Even without sharing, the Abuse delay attack would cause the Selection Algorithms to select very large suspicious collations, from the IIR , because of the noise made by the non-colluding (M)SPs, during the prolonged Abuse delay, rendering the whole GT testing useless, see Subsection 5.6.2 for more details about the IIR . For example, if the suspected collation size is 10, then the resulted TGT_{sg} Rank would be at least 80%, which indicates very good Trustworthiness.

The Strong Colluding Group Metric, TGT_{sc} , applies deeper analysis on the MGR entries to detect possible collations. That is, it only considers Popular SPs, PSPs, through the use of the SPR Record of Subsection 5.6.1, for its colluding analysis since MPSPs are the most influential partners in any collation because of their popularity among Users. Moreover, the TGT_{sc} considers the frequency of appearance for any recorded G_x including in the case where its members appear as members within a larger G_y . The two factors that would reduce the TGT_{sc} Rank for a G_x are: smaller G_x size and higher frequency of appearance. Once the combination of these two factors reaches a pre-adjusted threshold by the Auditor, the TGT_{sc} Rank would be 0. This threshold is called the Popular Colluding Record Threshold, or simply $pcrThreshold$. In a similar fashion to the TGT_{sg} Ranking Algorithm, there are $TGT_{sc}(SP)$ Ranks that are calculated using Equation 5.8 and stored in the $TGTR_{sc}(SP)$ Record. In addition, a modified version of the MGR Record called MGR_{sc} is created for the purpose of counting the frequency of appearances for each G_x even within a larger G_y .

$$\text{Given: } pcrNum(SP_x) = \sum_{i=0}^n \frac{2}{G_i.size()} * G_i.counter()$$

where n is the number of Gs containing SP_x ,

$G_i.size()$ is the counter of how many times G_i appears in the MGR ,

and $pcrNum$ is a variable trying to count the number of assured collations

i.e. spam received after just two SPs got the Credential

$$\text{Then: } TGT_{sc}(SP_x) = \frac{pcrThreshold - pcrNum(SP_x)}{pcrThreshold} * 100 \quad (5.8)$$

Note that $TGT_{sc}(SP_x)$ lower limit is 0

Equation 5.8: Strong Colluding Group Trust Rank, $TGT_{sc}(SP)$

5.4.4 Weak Colluding Group Metric

While TGT_{sc} excels at detecting colluding MPSPs that normally would bypass the TGT_{sg} Ranking Algorithm, by launching the Abuse Delay Attack of Subsection 5.8.6, there are cases where it would dramatically fail. Particularly speaking, when the colluding MPSPs realise how the TGT_{sc} works, they could deploy more complicated colluding policies to elude it, Subsection 5.8.7. For example, they could decide not to Abuse any User unless she deals with a minimum number of colluding MPSPs regardless of the used Credential, $N_{toAbuseU}$, to prove that it is a real User rather than a GT Agent with a single testing Credential. As an added advantage for the attackers, such a policy would enable them to identify the new User as a human targeted User in a faster way since all the Credentials she uses are counted in the decision process rather than focusing on only the interactions of 1 Credential at a time. Such a policy would trick the TGT_{sc} Ranking Algorithm to suspect innocent PSPs to be colluding because the reported Abuse logs for a single Abused Credential could simply include only one member of the collation while the rest of the collation members could have dealt with other Credentials generated by the Agent, to prove he is a genuine User.

In an attempt to counter such an attack, we introduce the Weak Colluding Group Metric, TGT_{wc} that is another Intermediate Ranking Algorithm for the TGT Ranking Approach. While the TGT_{wc} colluding analysis Equation is the same as TGT_{sc} , see Subsection 5.4.2, this Ranking Algorithm has two fundamental differences:

- The generated MGR_{wc} includes not only the PSPs that have interacted with the Abused Credential, but also the PSPs that have dealt with all the User's Credentials so far.
- To increase the maliciousness detection speed, TGT_{wc} does not only consider reports coming from GT Agents, but also all the $IILs$, see Subsection 5.6.2, coming from ordinary Users.

From the above-mentioned rules, it can be seen why this Algorithm is named Weak. It is normally unfair to consider every PSP that a User has dealt with prior to an

Abuse Case as a suspected collation member. In addition, it is quite possible that ordinary Users could act maliciously, perhaps artificial Users created by colluding entities, to destroy the integrity of the Trust Network. For that, this Algorithm should be deployed and utilized with extra caution to avoid mistakenly ranking down innocent PSPs. For that, a separate Popular Weak Colluding Record Threshold, *pcrWeakThreshold*, is created to control how harsh this Algorithm should be in deducing a PSP is in fact colluding.

5.4.5 Group Testing Metric Record

This is the Final Ranking Algorithm for the TGT Ranking Approach. It is dependent on the aggregated $TGT_{sg}(SP_x)$, $TGT_{sc}(SP_x)$, and $TGT_{wc}(SP_x)$ Ranks for each SP_x within the Network. The Auditor would give each of these three Intermediate Rankings a custom weight depending on the adopted Defensive Strategy, see Section 6.9, and the currently observed Network status, see Subsection 5.6.4, to generate the final $TGT(SP_x)$. All the different $TGT(SP)$ Ranks should be stored in a special Record called the Group Testing Trust Record, or simply *TGTR*. The entries of this Record can be calculated using the Equation 5.9.

$$TGT(SP_x) = W_{sg} * TGT_{sg}(SP_x) + W_{sc} * TGT_{sc}(SP_x) + W_{wc} * TGT_{wc}(SP_x) \quad (5.9)$$

$$\text{where: } W_{sg} + W_{sc} + W_{wc} = 100$$

Equation 5.9: Group Trust Rank, TGT(SP)

The case study 5.6 gives a glance at the main features and weaknesses of each Intermediate Ranking Algorithm. This case study shows how different *TGTRs* could be generated using the same Cases log based on the different assigned weights for each of the Intermediate Ranking Algorithms. In addition, it shows that the $TGT_{sg}(SP)$

would be very similar to the $TGT_{sc}(SP)$ because both are dependent on the reports coming only from GT Agents regarding PSPs that have dealt with the Abused Credential only. One main difference between the two Ranking Approaches is the fact that it is possible to reduce the Auditor’s certainty about the observed Strong Colluding Attack by increasing the $pcrThreshold$ value. In the case study, the $pcrThreshold$ is set to 2 causing the TGT_{sc} Rank of the innocent PSP_4^{DGU} to be better than its TGT_{sg} Rank since TGT_{sg} lacks the certainty adjustment feature. Another main difference, that cannot be spotted in this case study, is the fact that if the TGT_{sc} Ranking Algorithm detects a small suspicious G_x appearing together even once, it counts all the other appearances of G_x members together, even within larger Gs, to filter out any noise generated by normal Users’ activities and speed up the detection process.

When it comes to the TGT_{wc} Ranking Algorithm, the results are quite different and generally harsher than the rest of the TGT Intermediate Ranking Algorithms causing more malicious and even some innocent PSPs to be ranked malicious as shown in the case study. The reason is the fact that this Algorithm depends on the reports coming from all the Network Users regarding all the PSPs a single User has dealt with prior to receiving the first Abuse. This feature makes it possible to detect complicated colluding attacks where the attack would be spanning across many Users’ Credentials, see Subsections 5.4.4 and 5.8.7. In the other hand, TGT_{wc} is too aggressive since it would consider groups of PSPs to be colluding just because they appear frequently in the reported Cases. For that, the Auditor may decide to set the value of $pcrWeakThreshold$ a bit high to reduce the probability of ranking an innocent PSP as malicious.

Example 5.6: TGT Case Study

let the entities of the Trust Network look like Table 5.8, current Users interactions look like Table 5.9, current Agents interactions look like Table 5.10, $PCRThreshold == 2$, and $PCRWeakThreshold == 4$. Then, the generated MGR , MGR_{sc} , and MGR_{wc} would look like Tables 5.11, 5.12, and 5.13 respectively. Furthermore, the generated $TGT_{sg}(G)$, $TGT_{sg}(SP)$, $TGT_{sc}(SP)$, and $TGT_{wc}(SP)$ would look like Tables 5.14, 5.15, 5.16, and 5.17 respectively. Finally, Tables 5.18, 5.19, 5.20, and 5.21 show

the different generated *TGT* values depending on the distributed weights given to the different *TGT* Intermediate Ranking Algorithms. Note that the green cells denominate correct Ranks, red cells denominate wrong Ranks, and white cells denominate neutral Ranks. Also, note that in both Tables 5.9 and 5.10, the Abuse event is coloured the same as the SPs who caused it for illustration purposes. In real-life, the Auditor would have no idea who caused an Abuse or even how to differentiate for sure between PSPs, SPs, MSPs, or MPSPs.

SP	PSP	MSP	MPSP Colluding	MPSP Uncolluding
SP_1	PSP_4^{DGU}	MSP_2	$MPSP_3^{DGU}$	$MPSP_{13}$
SP_6	PSP_5	MSP_9	$MPSP_7$	
SP_{10}^{DGU}	PSP_8	MSP_{11}^{DGU}	$MPSP_{15}$	
SP_{12}	PSP_{14}	MSP_{17}		
	PSP_{16}			

TABLE 5.8: Trust Network Entities

	t_1	t_2	t_3	t_4	t_5	t_6
U_1						
Cred.1	PSP_4^{DGU}	SP_{12}	Abuse	$MPSP_7$	MSP_5	$MPSP_7$
PSP_4^{DGU} Shared with	$MPSP_{13}$					
G'_{wc}	$PSP_4^{DGU}, MPSP_{13}, PSP_{14}$					
Cred.2	PSP_{14}	MSP_2	PSP_8	$MPSP_3^{DGU}$		Abuse
$MPSP_3^{DGU}$ Shared with	MSP_{17}					
G'_{wc}	$MPSP_3^{DGU}, PSP_4^{DGU}, MPSP_7, PSP_8, MPSP_{13}, PSP_{14}$					
U_2						
Cred.1	PSP_{16}	$MPSP_{15}$	$MPSP_{13}$	Abuse	Abuse	SP_1
G'_{wc}	$MPSP_3^{DGU}, PSP_5, MPSP_{13}, MPSP_{15}, PSP_{16}$					
Cred.2	SP_{12}	$MPSP_3^{DGU}$		Abuse	MSP_9	
$MPSP_3^{DGU}$ Shared with	MSP_9, PSP_5					
G'_{wc}	$MPSP_3^{DGU}, PSP_5, MPSP_{13}, MPSP_{15}, PSP_{16}$					

TABLE 5.9: Current Users Interactions

	t ₁	t ₂	t ₃	t ₄	t ₅	t ₆
GT₁						
G	$PSP_4^{DGU}, MPSP_{13}, PSP_{16}$					
Cred.1	PSP_{16}	PSP_4^{DGU}	$MPSP_{13}$	Abuse	Abuse	
PSP_4^{DGU} Shared with	$MPSP_{15}$					
G'_{wc}	$PSP_4^{DGU}, MPSP_7, MPSP_{13}, PSP_{14}, MPSP_{15}, PSP_{16}$					
Cred.2 Unimportant	$MPSP_7$	PSP_{14}	SPAM	PSP_5		
G'_{wc}	$PSP_4^{DGU}, MPSP_7, PSP_{14}, MPSP_{15}, PSP_{16}$					
G'	$PSP_4^{DGU}, MPSP_{13}, MPSP_{15}, PSP_{16}$					
GT₂						
G	$MPSP_{13}, MPSP_{15}, PSP_{16}$					
Cred.1	$MPSP_{13}$	PSP_{16}	Abuse			
G'_{wc}	$PSP_4^{DGU}, MPSP_7, MPSP_{13}, PSP_{16}$					
Cred.2 Unimportant	PSP_4^{DGU}	$MPSP_7$	PSP_8			
PSP_4^{DGU} Shared with	SP_{12}					
G'_{wc}						
G'	$MPSP_{13}, PSP_{16}$					

TABLE 5.10: Current Agents Interactions

MGR	
G	Count
$PSP_4^{DGU}, MPSP_{13}, MPSP_{15}, PSP_{16}$	1
$MPSP_{13}, PSP_{16}$	1

TABLE 5.11: MGR

MGR_{sc}	
G	Count
$PSP_4^{DGU}, MPSP_{13}, MPSP_{15}, PSP_{16}$	1
$MPSP_{13}, PSP_{16}$	2

TABLE 5.12: MGR_{sc}

MGR_{wc}	
G	Count
$PSP_4^{DGU}, MPSP_{13}, PSP_{14}$	3
$MPSP_3^{DGU}, PSP_4^{DGU}, MPSP_7, PSP_8, MPSP_{13}, PSP_{14}$	1
$MPSP_3^{DGU}, PSP_5, MPSP_{13}, MPSP_{15}, PSP_{16}$	2
$PSP_4^{DGU}, MPSP_7, MPSP_{13}, PSP_{14}, MPSP_{15}, PSP_{16}$	1
$PSP_4^{DGU}, MPSP_7, PSP_{14}, MPSP_{15}, PSP_{16}$	2
$PSP_4^{DGU}, MPSP_7, MPSP_{13}, PSP_{16}$	2

TABLE 5.13: MGR_{wc}

TGT_{sg}(G)	
G	Rank
$PSP_4^{DGU}, MPSP_{13}, MPSP_{15}, PSP_{16}$	50%
$MPSP_{13}, PSP_{16}$	0%

TABLE 5.14: TGT_{sg}(G)

TGT_{sg}(SP)	
SP	Rank
$MPSP_3^{DGU}$	100%
PSP_4^{DGU}	50%
PSP_5	100%
$MPSP_7$	100%
PSP_8	100%
$MPSP_{13}$	0%
$MPSP_{15}$	50%
PSP_{16}	0%

TABLE 5.15:
TGT_{sg}(SP)

TGT_{sc}(SP)	
SP	Rank
$MPSP_3^{DGU}$	100%
PSP_4^{DGU}	75%
PSP_5	100%
$MPSP_7$	100%
PSP_8	100%
$MPSP_{13}$	-25% ⇒ 0%
$MPSP_{15}$	75%
PSP_{16}	-25% ⇒ 0%

TABLE 5.16:
TGT_{sc}(SP)

TGT_{wc}(SP)	
SP	Rank
$MPSP_3^{DGU}$	71.67%
PSP_4^{DGU}	-11.67% ⇒ 0%
PSP_5	80%
$MPSP_7$	38.33%
PSP_8	91.67%
$MPSP_{13}$	-11.67% ⇒ 0%
$MPSP_{15}$	51.67%
PSP_{16}	26.67%

TABLE 5.17:
TGT_{wc}(SP)

TGT	
TGT_{sg} weight	33.33%
TGT_{sc} weight	33.33%
TGT_{wc} weight	33.33%
SP	Rank
$MPSP_3^{DGU}$	90.55%
PSP_4^{DGU}	41.66%
PSP_5	93.32%
$MPSP_7$	79.44%
PSP_8	97.21%
$MPSP_{13}$	0%
$MPSP_{15}$	58.88%
PSP_{16}	8.89%

TABLE 5.18: TGT Weights Combinations 1

TGT	
TGT_{sg} weight	80%
TGT_{sc} weight	10%
TGT_{wc} weight	10%
SP	Rank
$MPSP_3^{DGU}$	97.17%
PSP_4^{DGU}	47.50%
PSP_5	98%
$MPSP_7$	93.83%
PSP_8	99.17%
$MPSP_{13}$	0%
$MPSP_{15}$	52.67%
PSP_{16}	2.67%

TABLE 5.19: TGT Weights Combinations 2

TGT	
TGT _{sg} weight	10%
TGT _{sc} weight	80%
TGT _{wc} weight	10%
SP	Rank
<i>MPSP</i> ₃ ^{DGU}	97.17%
<i>PSP</i> ₄ ^{DGU}	65%
<i>PSP</i> ₅	98%
<i>MPSP</i> ₇	93.83%
<i>PSP</i> ₈	99.17%
<i>MPSP</i> ₁₃	0%
<i>MPSP</i> ₁₅	70.17%
<i>PSP</i> ₁₆	2.67%

TABLE 5.20: TGT Weights Combinations 3

TGT	
TGT _{sg} weight	10%
TGT _{sc} weight	10%
TGT _{wc} weight	80%
SP	Rank
<i>MPSP</i> ₃ ^{DGU}	77.34%
<i>PSP</i> ₄ ^{DGU}	12.5%
<i>PSP</i> ₅	84%
<i>MPSP</i> ₇	50.66%
<i>PSP</i> ₈	93.34%
<i>MPSP</i> ₁₃	0%
<i>MPSP</i> ₁₅	53.84%
<i>PSP</i> ₁₆	21.34%

TABLE 5.21: TGT Weights Combinations 4

5.5 Global Trust Metric Record

The Global Trust Metric, $TG(SP)$, is the Final Ranking Algorithm that incorporates all of the presented Ranking Approaches as they appear in Figure 5.1. It is dependent on the aggregated $TL_{avg}(SP)$, $TST(SP)$, and $TGT(SP)$ Ranks for each SP_x within the Network. The Auditor would give each of those three Ranks a custom weight depending on the adopted Defensive Strategy, see Section 6.9, and the currently observed Network status, see Subsection 5.6.4, to generate the final $TG(SP)$. All the different $TG(SP)$ Ranks should be stored in a special Record called the Global Trust Metric Record, or simply TGR . The entries of this Record can be calculated using the Equation 5.10 while Example 5.7 demonstrate how it works.

Note that it may seem counter intuitive to assume that, for a certain SP_x , it is possible to get $TL_{avg}(SP_x) = null$ while $TST(SP_x) \neq null$ since no SP would be added to the Auditor ranking system before it appears in a reported Case and each reported Case would be processed by both the GPD and the STD Algorithms which, in turn, produces the TL_{avg} and TST Ranks. However, when an SP_x is banned from the system, see Section 5.7, it is possible to ignore all the Cases where SP_x appears since it would be the accused SP for Abuse in all those Cases. Such ignorance could be applied by all the Ranking Algorithms or might be restricted to only one Algorithm,

the TLR_{avg} for example. That is because ignoring all the Cases where an old MPSP has appeared for a long time would cause a massive number of logs to be ignored causing a major re-ranking shake to the system and increased inaccuracy that could be irrecoverable, see Subsection 5.8.8.

$$TG(SP_x) = W_{TL_{avg}} * TL_{avg}(SP_x) + W_{TST} * TST(SP_x) + W_{TGT} * TGT(SP_x) \quad (5.10)$$

: where if any ranking value is null, its W would be divided equally on the remaining ranks

Equation 5.10: Global Trust Metric, TG

Example 5.7: Generating the TGR Record

Assume that at the current moment the TLR_{avg} Record looks like Table 5.22, the $TSTR$ Record looks like Table 5.23, and the $TGTR_{SP}$ Record looks like Table 5.24. If we let the Ranks weights to be $W_{TL_{avg}} = 10$, $W_{TST} = 20$, and $W_{TGT} = 70$, then the corresponding TGR Record would look like Table 5.25.

SP	$TL_{avg}(SP)$
SP_1	50%
SP_2	70%
SP_3^{DGU}	100%
SP_4	44%
SP_5	20%
SP_6	null
SP_7^{DGU}	60%
SP_8	19%
SP_9	null

TABLE 5.22: TLR_{avg}

SP	TST(SP)
SP_1	null
SP_2	85%
SP_3^{DGU}	100%
SP_4	20%
SP_5	38%
SP_6	null
SP_7^{DGU}	40%
SP_8	47%
SP_9	null

TABLE 5.23: TSTR

SP	$TGT_{SP}(SP)$
SP_1	36%
SP_2	44%
SP_3^{DGU}	100%
SP_4	59%
SP_5	null
SP_6	60%
SP_7^{DGU}	75%
SP_8	0%
SP_9	null

TABLE 5.24: $TGTR_{SP}$

SP	TG(SP)
SP_1	33.8%
SP_2	54.8%
SP_3^{DGU}	100%
SP_4	49.7%
SP_5	29.9%
SP_6	60%
SP_7^{DGU}	66.5%
SP_8	11.3%
SP_9	null

TABLE 5.25: TGR

5.6 Utility Units

In this Section, we describe the basic Utility Units and how they would work as they appear in Figure 5.1. In a nutshell, The SPs Popularity Record, *SPR*, is a Selection Algorithm that aid the Auditor in the task of selecting candidate SPs for ST or GT Raw Agent Testing based on their popularity among Users. The Initial Interaction Record, *IIR*, is a Record of all the Initial Interactions Lists, *ILL*, for the Users that are part of the Trust Network and would also serve as a Selection Algorithm since the first SPs to acquire the compromised Credential are probably the ones who are continuing to compromise it. The Data Governance Unit, *DGU*, is a powerful Utility Unit to enforce Hard Trust in the Trust Network. The Trust Network Health Gauges are a set of sensors to update the Auditor about the current status of the Network and its own performance to adjust its defensive parameters, if necessary.

5.6.1 *SPs Popularity Record*

The ST Ranking Agent, see Subsection 5.3.2, is vital to detect malicious SPs that operate independently while the GT Ranking Agent, see Subsection 5.4.1, does a great job in detecting potential collusion among MSPs within the managed Trust Network. Nevertheless, there must be some efficient and effective Selection Algorithms to select single SPs for ST testing or groups of them for GT testing. The SPs Popularity Record, *SPR*, is one of the vital Selection Algorithms to aid the Auditor in the task of selecting candidate SPs for ST or GT testing.

The *SPR*, is necessary because most of the Users would deal most of the times with a small set of popular SPs, PSPs, like Google, Facebook, and Twitter, for example. The presence of other insignificant SPs in most of the reported Cases would generate Data noise diverging the focus of the Ranking Algorithms away from the significant (M)PSPs. For this, the *SPR*, should be created to order the SPs based on their popularity. Then, Auditor could identify the Popular SPs List, *PSL*, by considering the the top %20 of SPs in the *SPR*. This list could be then the subject for further investigations, depending on the Deployment Rules of the Auditor, see Section 5.7.

For the sake of simplicity, the %20 threshold is chosen based on the Pareto Principle, which states that %80 of effects is generated by %20 of causes [121]. Nevertheless, the Auditor admin could decide to change this threshold if necessary. Algorithm 5.5 shows the procedure to populate the *SPR* while Example 5.8 demonstrates it.

Algorithm 5.5: The SPs Popularity Record, SPR

- At the initialization of the Trust Network, create an empty *SPR* Record.
- For every newly reported case C_i , traverse each $SP \in C_i$.
- If the currently traversed SP does not have an entry in the *SPR*, create a new entry for it with a counter value = 1.
- If the currently traversed SP has an entry in the *SPR*, increase its counter by 1.
- Keep the *SPR* ordered by the value of its counters.
- If the current *PSL* is requested, return the top %20 of the *SPR*.

Example 5.8: Generating the SPR Record

If the current recorded activities by all Users looks like Table 5.26, then the *SPR* Record would look like Table 5.27.

User	Interactions
U_1	$SP_1, SP_3^{DGU}, Abuse!, SP_5, SP_1, Abuse!$
U_2	$SP_3^{DGU}, SP_1, SP_4, Abuse!, SP_6, SP_1, Abuse!$

TABLE 5.26: Users' Interactions

<i>PSL?</i>	SP	Counter
<i>PSL</i>	SP_1	4
	SP_3^{DGU}	2
	SP_4	1
	SP_5	1
	SP_6	1

TABLE 5.27: *SPR*

5.6.2 Initial Interactions Record

As stated in 5.6.1, there must be some efficient and effective Selection Algorithms to select groups of SPs for *ST* and *GT* Raw Testing Agents inspections. The Initial Interactions Record, IIR, is another powerful Selection Algorithm to aid the Auditor in that selection task. The *IIR* is a Record of all the Initial Interactions Lists, IIL,

for the Users that are part of the Trust Network. The *IIL* is a good starting point for the Auditor to form a testing group, G_i , that is a good candidate for further GT Agents testing. That is because if U_x has dealt with G_i members before getting its first Abuse, then the Auditor would be sure that at least one $SP_x \in G_i$ is acting maliciously. If U_x continued interacting with G_{i+1} before getting the second Abuse, it would be wrong to assume that there is at least one $SP_y \in G_{i+1}$ and $SP_y \notin G_i$ acting maliciously. That is because the Abuse might be coming from the malicious $SP_x \in G_i$ that we assumed its existence earlier.

In real-life large Trust Networks, the raw *IIR* would still be a large inefficient pool of Gs to choose from for GT Agents testing. That is, fine grained selection mechanisms are needed. For that, we create a statistical Record named the Popular IILs, *PIILs*. As the name suggests, this Record would count how many times a certain *IIL* is observed within the *IIR* to select the mostly reported *IILs*. That is important because those frequently appearing SPs in the reported initial Abuses would be highly suspicious for engaging in colluding activities. The chronological order of SPs in *IILs* should be neglected by both the *IIR* and the *PIILs*. Moreover, if $IIL_x \in IIL_y$, then the IIL_y should be counted twice: in its own Record and in the IIL_x Record because members of IIL_x might be the main colluding party in the whole Trust Network and the presence of other SPs with them in other *IILs* might be just coincidences, i.e. Data noise. The *IILs* entries of the *PIILs* Record should be ordered by the highest appearing frequency. Algorithm 5.6 shows the procedure to populate the *IIR* and its

associated *PIILs* while Example 5.9 demonstrates it.

Algorithm 5.6: The Initial Interactions Record, *IIR* and The *PIILs*

- At the initialization of the Trust Network, create an empty *IIR* Record and an empty *PIILs* Record.
- If a U_x reports an Abuse to a Credential $Cred_y$, check whether $Cred_y$ has a presence in the *IIR*, i.e. whether this is the initial Abuse to it.
- If the previous check reveals that this is the initial Abuse to $Cred_y$, then create a new *IIL* for $Cred_y$, IIL_y , containing all the unique SPs that U_x has dealt with using $Cred_y$ prior to getting its initial Abuse.
- Attach to the IIL_y a list of PSPs, categorised by the *SPR* of Subsection 5.6.1, that U_x has dealt with using all its Credentials prior to getting $Cred_y$ initial Abuse. This list is needed by the TGT_{wc} Ranking Algorithm of Subsection 5.4.4. Name this list G_{wc} .
- Add IIL_y to the *IIR*.
- Check whether the members of IIL_y , G_v , exist in the *PIILs* as a standalone entry. If it does, then increase that entry's Counter by 1.
- If the previous check reveals that G_v does not exist in the *PIILs*, then add IIL_y to the *PIILs*.
- When adding a new IIL_y to the *PIILs*, check all the *PIILs* entries. If an entry IIL_z where $IIL_y \in IIL_z$ exists, then increase the IIL_y counter by 1. If an entry IIL_w where $IIL_w \in IIL_y$ exists, then increase IIL_w counter by 1.
- After adding a new IIL_y to the *PIILs*, update the *PIILs* entries on a descending order based on the entries counters.

Example 5.9: Generating the IIR Record

If the current Users' interactions look like Table 5.28, then the generated *IIR* should look like Table 5.29 and the associated *PIILs* should look like Table 5.30.

	t_1	t_2	t_3	t_4	t_5	t_6
U_1						
C_1	PSP_3^{DGU}	PSP_{13}	Abuse	PSP_7	SP_5	PSP_{10}
PSP_3^{DGU} Shared with	PSP_8					
C_2	PSP_{14}	SP_2	PSP_3^{DGU}	PSP_8	PSP_4^{DGU}	Abuse
PSP_3^{DGU} Shared with	SP_{17}					
PSP_4^{DGU} Shared with	PSP_{13}					
U_2						
C_1	PSP_{14}	PSP_8	PSP_{13}	Abuse	SP_1	Abuse
C_2	PSP_8	PSP_3^{DGU}		Abuse	SP_9	
PSP_3^{DGU} Shared with	PSP_{13}					

TABLE 5.28: Current Users Interactions

List	Contents
$IIL_{U_1:C_1}$	$PSP_3^{DGU}, PSP_8, PSP_{13}$
$G_{wc}(IIL_{U_1:C_1})$	$PSP_3^{DGU}, PSP_8, PSP_{13}, PSP_{14}$
$IIL_{U_1:C_2}$	$SP_2, PSP_3^{DGU}, PSP_4^{DGU}, PSP_8, PSP_{13}, PSP_{14}, SP_{17}$
$G_{wc}(IIL_{U_1:C_2})$	$PSP_3^{DGU}, PSP_4^{DGU}, PSP_7, PSP_8, PSP_{13}, PSP_{14}$
$IIL_{U_2:C_1}$	$PSP_8, PSP_{13}, PSP_{14}$
$G_{wc}(IIL_{U_2:C_1})$	$PSP_3^{DGU}, PSP_8, PSP_{13}, PSP_{14}$
$IIL_{U_2:C_2}$	$PSP_3^{DGU}, PSP_8, PSP_{13}$
$G_{wc}(IIL_{U_2:C_2})$	$PSP_3^{DGU}, PSP_8, PSP_{13}, PSP_{14}$

TABLE 5.29: IIR

List	Contents	Count
IIL_1	$PSP_3^{DGU}, PSP_8, PSP_{13}$	3
IIL_2	$PSP_8, PSP_{13}, PSP_{14}$	2
IIL_3	$SP_2, PSP_3^{DGU}, PSP_4^{DGU}, PSP_8, PSP_{13}, PSP_{14}, SP_{17}$	1

TABLE 5.30: PIILs

5.6.3 Data Governance Unit

The Data Governance Unit, DGU, is a powerful Utility Unit to enforce Hard Trust in the Trust Network. As detailed in Section 4.4, the DGU Unit should be implemented so that it makes it impossible for any SP that acquires a Credential to handle it without compliance to the Credential's Sticky Policies. In addition, DGU should record all internal accesses to the Credential as well as all its approved sharing destinations. However, deploying DGU without proper Deployment Rules and a stable supporting Ranking Algorithms would not guarantee Trust Enforcement within the Trust Network, see Subsection 2.8. That is mainly because it is not expected that all Network entities, Users and SPs, would agree to *Install DGU* once it is rolled out without initial hesitation and resistance. Hence, there would still be loopholes during the bootstrapping period where MSPs would be able to Abuse whatever Credentials they acquire by sharing them, with compliance to the Sticky Policies, to other MSPs, which did not *Install DGU* yet, to do the Abuse on their behalf, see Subsection 5.8.7. Of course, targeted attacks is always possible, see Section 2.4, but dealing with this category of malicious attacks is out of this thesis scope, see Section 1.3.

Given the associated vulnerabilities of deploying DGU during bootstrapping period when not all entities are expected to voluntarily *Install DGU*, there are some rules the Auditor can customise to mitigate this issue. First of all, when the Rank of a given SP gets lower than the Banning-Threshold, see Section 5.7, it could give that SP the chance to avoid getting banned by agreeing to *Install DGU*. This would help in reducing the amount of banned innocent SPs by giving them the chance to prove their innocence. Nevertheless, those innocent SPs may find it annoying to *Install DGU* with all its limitations, specially in small new Trust Networks that are trying to bootstrap. For such SPs, it might be preferable to be banned instead of *Installing DGU*. Hence, the following two installations options could be utilized by the Auditor to make it easier for SPs to start adopting the DGU technology:

- **Install DGU:** this option allows the SP that agrees to *Install DGU* to share the acquired Credentials with any other SP as long as this sharing does not

violate the Credentials' Sticky Policies.

- **Enable Strict DGU:** in addition to satisfying the Credentials' Sticky Policies, this option restricts the SP that has already *Installed DGU* from sharing the acquired Credentials unless with other SPs that have also *Installed DGU* to completely close the loophole that enables Data abusing by sharing with SPs that have not *Installed DGU* yet.

Given the above installation options, the Auditor should deploy different handling rules for each SP *Installing DGU* depending on the installation option. It should be safe to give an SP that choose to *Enable Strict DGU* option a 100% Trust Rank and, then, start ignoring its appearance in new Cases' logs since it won't be able to Abuse Users under any circumstances, except for Targeted Attack Abuse that is out of scope. On the other hand, SPs that choose the *Install DGU* option should not get absolute Trust. Rather, they should be ranked and evaluated by the ordinary Ranking Algorithms just like any other SP. However, once an SP choose to *Install DGU*, it should get a fresh start by clearing its $TLLR_{avg}$, TST , and TGT Ranks. In addition, it should get a new high TGR Rank as an appreciation for their acceptance to *Install DGU*. The colluding SPs which were forced to *Install DGU* may resort to more complicated forms of colluding that are more challenging to detect by the current Ranking Algorithms, see Subsection 5.8.7. For that, the *Post_DGU_Install-Rank* should not be too high, so it does not take too long to lower down and, eventually, detect its malicious behaviour, if there is any. Algorithm 5.7 explains how an Auditor

could utilize the DGU.

Algorithm 5.7: How Auditor could Utilize DGU

- If $TGR(SP_x) \leq Banning_Threshold$, then the Auditor could offer SP_x to either *Install DGU* or getting banned of the Trust Network
- If SP_x accepted to *Install DGU*, then:
 - Ignore all its $TLLR_{avg}$ Cases prior to this time without deleting or ignoring the $TLLR_{avg}$ Ranks for other SPs that appeared with SP_x in the same logs.
 - Delete all its *TST* and *TGT* Cases, including the rest of SPs that appear within those Cases.
 - Give SP_x a new high TGR Rank = *Post_DGU_Install_Rank*.
 - Continue evaluating SP_x as normal.
- If the TGR Rank of $SP_x^{DGU} \leq Banning_Threshold$, then the Auditor should offer SP_x^{DGU} to either *Enable Strict DGU* or getting banned from the Network.
- If SP_x accepted to *Enable Strict DGU*, then:
 - Ignore all its $TLLR_{avg}$ Cases prior to this time without deleting or ignoring the $TLLR_{avg}$ Ranks for other SPs that appeared with SP_x in the same logs.
 - Delete all its *TST* and *TGT* Cases, including the rest of SPs that appear within those Cases.
 - Give SP_x a new high TGR Rank = *Post_Enable_Strict_DGU_Rank* (normally 100%).
 - Stop evaluating SP_x and ignore all its appearances in new Cases.
- If SP_x denied to *Install DGU* or *Enable Strict DGU* when either is offered, then:
 - Close all the Cases where SP_x appears by declaring that SP_x is accused as the Abuser.
 - Depending on the Deployment Rules, the Auditor may want to ignore all the *TGT* logs where SP_x has appeared.
 - Ban SP_x from the Trust Network.

5.6.4 Trust Network Health Gauges

In a dynamic large Trust Network, it should be expected that Users would have altering interaction trends while the malicious entities would deploy adaptive attacks in their quest to bypass the proposed Ranking Algorithms and Defensive Strategies, see Section 5.8. For that, it is crucial for a successful Trust Auditor to have a set of powerful sensors to provide feedback reports about the health status of the Trust Network. Such sensors, which we call Trust Health Gauges, would let the Auditor know whether its current deployed Defensive Strategy is doing a good job or it is time to try something new since the attackers may have already optimized their Counter-Attacking Strategies, see Section 6.7 for details about the Defensive Strategies and the Counter-Attacking Strategies.

Four general groups of Trust Health Gauges are described below. Note the omitting of the detailed equations since they are simple formulas that could be implied from the textual description. Also, note that a fifth possible group to analyse the malicious Users trends is not listed here since detecting malicious Users would require a User Reliability Index, see Subsection 7.3.7, which is out of this thesis scope.

Group 1: General Health Gauges

- The Trust Health Ratio of MSPs: $H_{sp} = 1 - \frac{SP_nMSPs-est}{SP_n}$
- The Trust Health Ratio of Undetected MSPs: $H_{sp-undetected} = \frac{SP_nMSPs-detected}{SP_nMSPs-est}$
- The Trust Health Ratio of MSPs Growth Rate: $H_{sp-GR} = 1 - \frac{SP_{GR-MSPs}}{SP_{GR}}$
- The Trust Health Ratio of Users Credentials Integrity: $H_{Cred} = 1 - \frac{Cred_n-abused}{Cred_n}$
- The Trust Health Ratio of Open Cases: $H_{C-open} = 1 - \frac{C_{open}}{C_n}$
- The Cases Growth Rate: $C_{GR}(t_2) = \frac{C_n(t_2)-C_n(t_1)}{C_n(t_1)*(t_2-t_1)}$
- The Cases with new PII Growth Rate:

$$C_{GR-newPII}(t_2) = \frac{C_{n-newPII}(t_2)-C_{n-newPII}(t_1)}{C_{n-newPII}(t_1)*(t_2-t_1)}$$
- The Users Growth Rate: $U_{GR}(t_2) = \frac{U_n(t_2)-U_n(t_1)}{U_n(t_1)*(t_2-t_1)}$
- The SPs Growth Rate: $SP_{GR}(t_2) = \frac{SP_n(t_2)-SP_n(t_1)}{SP_n(t_1)*(t_2-t_1)}$
- The MSPs Growth Rate: $SP_{GR-MSPs}(t_2) = \frac{SP_nMSPs-est(t_2)-SP_nMSPs-est(t_1)}{SP_nMSPs-est(t_1)*(t_2-t_1)}$

The metrics of this group could be used by the Auditor to imply many facts about the current health status of the Trust Network. For example, a low H_{sp} would indicate an infected Network with a high density of MSPs and, hence, Users should take extra caution when dealing with any SP while the Auditor should deploy more Aggressive Defensive Strategies, see Section 6.7. Moreover, a high H_{sp-GR} value would indicate that the Network is getting more attractive for attackers. That could be caused by its weak Defensive Strategy, its huge success, or both. Another example would be a large value of H_{Cred} combined with a small value of H_{C-open} would indicate a low volume of malicious activities that are complex enough to defeat the Defensive Strategy. That would be an indicator that the current deployed Defensive Strategy has successfully deterred malicious entities from launching inexpensive massive attacks and resorted to more expensive and less attractive forms of colluding attacks as the only way to gain anything from the Trust Network, leading to a small number of Abuse Cases that are mostly not resolved. A final example would be the observation of large values of U_{GR}

and SP_{GR} that would indicate an active and growing Network while small values of the same variables would indicate a saturating, possibly distrusted, Network status.

Group 2: Testing Agents Health Gauges

- The Performance Ratio of the ST Agents: $P_{ST} = \frac{ST_+}{ST_n}$ where ST_+ is the number of Agents that received Abuse, as anticipated by the Selection Algorithm
- The Performance Ratio of the GT Agents: $P_{GT} = \frac{GT_+}{GT_n}$ where GT_+ is the number of Agents that received Abuse, as anticipated by the Selection Algorithm
- The MIN, MAX, AVG, and Deviation σ of the Following Variables: $Size(GT_+)$, τ_{toST_+} , τ_{toGT_+} , N_{toST_+} (number of recreated STs needed to detect an MSP), and N_{toGT_+} (number of recreated GT Agents to detect a malicious collation)

These internal Health Gauges should help the Auditor in optimising how the Raw Ranking Agents are deployed. For example, large P_{ST} and P_{GT} values may indicate effective Selection Algorithms and, hence, the Auditor may decide it is not yet the time to change the settings (unless the G sizes for the GT Agents is very large, leading to less assertive Ranks). Another example is utilising the AVG value of $Size(GT_+)$ to optimize the selected G size for new GT Agents while the values of τ_{toST_+} and τ_{toGT_+} could be used to adjust the Agents killing time. Furthermore, large N_{toST_+} and N_{toGT_+} would indicate the utilisation of a Users-Drop Attacking Algorithm by the attackers, see Subsection 5.8.6. Lastly, observing the deviation σ is vital because high σ values would indicate that the malicious activities do vary in their patterns and, hence, the AVG values should not be relied upon heavily to optimize the deployed Defensive Strategy.

Group 3: Malicious SPs Trends Gauges

- The Number of Actual Detected MSPs: $SP_{nMSPs-act}$
- The Number of Estimated Total MSPs: $SP_{nMSPs-est} = \frac{C_{open}}{AVG(C_{MR})} + SP_{nMSPs-act}$
- The number of Cases required to detect an MSP: $C_{MR}(SP_x) = \sum_{i=1}^n i$ where n is the number of all C_i such that $SP_x \in C_i$
- The MIN, MAX, AVG, and Deviation σ of the following variables: C_{MR} , $N_{MSPs-Uncolluding}(C_x)$ (number of suspicious individual SPs, not part of a suspected collations within C_x), and $N_{MSPs-Colluding}(C_x)$ (number of different suspicious malicious collations within C_x)

This is another set of internal Health Gauges to help the Auditor understands the MSPs trends and, hence, adjust its Defensive Strategies. For example, a low AVG value of C_{MR} could indicate an effective Defensive Strategy or, more likely, most of the detected attacks are made by immature attackers. However, combining a low AVG C_{MR} with a low $H_{sp-undetected}$ value would indicate that the current Defensive Strategy is only good at catching simple attacks and is ineffective at detecting more complicated attacks that are currently being launched. Another example would be observing high values of $N_{MSPs-Uncolluding}(C_x)$ and $N_{MSPs-Colluding}(C_x)$ indicating that there are many attacking entities, larger collation that are unobserved yet, or high ratio of Users not reporting all their Abuse Cases.

Group 4: Ranking Algorithms Trends Gauges

- The Ratio of detected SPs by each Ranking Algorithm:

$$R_{DetectedBy-TLR_{avg}}, R_{DetectedBy-TST}, R_{DetectedBy-TGT_{sg}}, R_{DetectedBy-TGT_{sc}}, \\ R_{DetectedBy-TGT_{wc}}, R_{DetectedBy-TGR}$$

- The Ratio of detected Colluding, Uncolluding, Banned, Installed DGU, and Enabled Strict DGU SPs: $R_{Detected-Colluding}, R_{Detected-Uncolluding}, R_{Banned}, R_{Install_DGU}, R_{Enable_StrictDGU}$

- The Growth Rate of detected MSPs by each Ranking Algorithm:

$$GR_{DetectedBy-TLR_{avg}}, GR_{DetectedBy-TST}, GR_{DetectedBy-TGT_{sg}}, GR_{DetectedBy-TGT_{sc}}, \\ GR_{DetectedBy-TGT_{wc}}, GR_{DetectedBy-TGR}$$

- The Growth Rate of detected Colluding, Uncolluding, Banned, Installed DGU, and Enabled Strict DGU SPs: $GR_{Detected-Colluding}, GR_{Detected-Uncolluding}, GR_{Banned}, GR_{Install_DGU}, GR_{Enable_StrictDGU}$

This is another set of internal Health Gauges to help the Auditor in understanding how its deployed Ranking Algorithms are performing and how the SPs are affected to better adjust its Defensive Strategies. For example, a low $R_{DetectedBy-TST}$ could indicate that most attackers at the moment are colluding entities and, hence, the ST Ranking Agent could be turned off to reduce the associated cost of running it. A high $R_{DetectedBy-TGT_{wc}}$ could indicate either more complicated forms of Colluding Attacks that are being deployed against the system or the fact that the currently deployed Defensive Strategy is so aggressive that many innocent SPs are being mistakenly detected by the TGT_{wc} Ranking Algorithm. The GR rates would also help the Auditor in adjusting the Algorithms' weights in the TG formula in favour of the Detector that is showing a higher GR rate. While high Ratios of $R_{Install_DGU}$ and $R_{Enable_Strict_DGU}$ would indicate that most of the SPs have finally accepted the DGU technology and, hence, the Abuse rates should be anticipated to drop down sharply. A high Ratio of R_{Banned} may indicate that the current Defensive Strategy is aggressive and many innocent SPs are refusing to *Install DGU* and prefer to be banned from the Network.

5.7 Basic Deployment Rules

By the aid of the aforementioned Ranking Algorithms and Utility Units, the Auditor should define a set of Deployment Rules to control how those basic building blocks would interact with each other. Below is a sample set of such rules that are suggested to start running the Trust Framework. Nevertheless, our exhaustive Simulation Process generated a set of Defensive Strategies with differing Deployment Rules suitable for different environmental conditions against a variety of Counter-Attacking Strategies, see Section 6.7.

General Rules

- **Rule 1:** Set $suspiciousSPRank = 25\%$.
- **Rule 2:** Create the $suspiciousSPs$ Record to contain all the SPs that show suspicious behaviour.
- **Rule 3:** Create the $bannedSPs$ Record to contain all the banned SPs from the Network.
- **Rule 4:** Set $banningSPRank = 5\%$.
- **Rule 5:** Set the $suspiciousRange = 5\%$.
- **Rule 6:** When we get $TG(SP_x) \leq suspiciousSPRank$ then add SP_x to the $guiltySPs(C_y) \forall C_y$ where $SP_x \in C_y$ and $\nexists SP_z \in C_y$ where $TG(SP_z) \leq TG(SP_x)$. Also, add SP_x to the $suspiciousSPs$ Record.
- **Rule 7:** $\forall C_x$ where $\exists SP_y \in guiltySPs(C_x)$ and $\exists SP_z$ where $TG(SP_z) \approx TG(SP_y) \pm suspiciousRange$, add $\forall SP_z$ to $guiltySPs(C_x)$ and to the $suspiciousSPs$ Record.
- **Rule 8:** $\forall C_x$ where $\exists SP_y$ where $TG(SP_y) \leq banningSPRank$, then give SP_y the option to *Install DGU*, *Enable Strict DGU* if it has already *Installed DGU*, or getting banned. If SP_y refused the DGU offer, add SP_y to the $bannedSPs$ Record and ban it from the Network.

- **Rule 9:** $\forall C_x$ where $MIN(guiltySPs(C_x)) = SP_y$ and $\exists SP_z \in guiltySPs(C_x)$ where $TG(SP_z) > (TG(SP_y) + suspiciousRange)$, then remove $\forall SP_z$ from $guiltySPs(C_x)$. If C_x has been already closed, then reopen it to update the $guiltySPs(C_x)$ Record and regenerate all the Ranks accordingly. If SP_z was banned, the ban might be reversed if the Ranks recalculation would lead to $TG(SP_z) > banningSPRank$.
- **Rule 10:** Create the *PSL* Record by aggregating the top 20% of the *SPR* entries.

TLR_{avg} Rules

- **Rule 1:** Set the minimum number of Users who have interacted with SP_x before deciding to ban SP_x based solely on $TLR_{avg}(SP_x)$ Ranking, in the absence of $TGT(SP_x)$ Rank, $TLRUsToBanSP = 5$.
- **Rule 2:** Set the minimum number of Users who have interacted with PSP_x before deciding to ban PSP_x based solely on $TLR_{avg}(PSP_x)$ Rank, in the absence of the $TGT(PSP_x)$ Rank, $TLRUsToBanPSP = 20$.

TST Rules

- **Rule 1:** \forall ST Agents, let $\tau_{ST-maxLife} = 60$ Days and $\tau_{ST-idle} = 1$ Day.
- **Rule 2:** Create an $ST(SP_x) \forall SP_x \in PSL$.
- **Rule 3:** Create an $ST(SP_x) \forall SP_x \in$ top 20% of TLR_{avg} .
- **Rule 4:** Create an $ST(SP_x) \forall SP_x \in$ bottom 20% of TLR_{avg} .
- **Rule 5:** Create an $ST(SP_x) \forall SP_x \in IIR(IIL_y) : IIL_y(size) = 1$.

TGT Rules

- **Rule 1:** \forall GT Agents, let $\tau_{GT-maxLife} = 60$ Days and $\tau_{GT-idle} = 1$ Day.

- **Rule 2:** Create a $GT(SP_x) \forall SP_x \in PSL$.
- **Rule 3:** Create a $GT(SP_x) \forall$ top 20% of $PIILs$ entries.
- **Rule 4:** Ignore $\forall G_y$ where $SP_x \in G_y$ and $SP_x \in bannedSPs$ in the TGT calculations. Recalculate all the TGT Ranks for the rest of SPs accordingly.

DGU Rules

- **Rule 1:** Create the *Installed DGU* Record to contain all the SPs that agreed to *Install DGU*.
- **Rule 2:** Create the *Enabled Strict DGU* Record to contain all the SPs that agreed to *Enable Strict DGU*.
- **Rule 3:** Set the $postInstallDGURank = 80\%$.
- **Rule 4:** Set the $postEnableStrictDGURank = 100\%$.
- **Rule 5:** Set the probability of inviting a random SP_x to *Install DGU* at a given day to be $vInstall DGU = 2\%$.
- **Rule 6:** Set the probability of inviting a random SP_x to *Enable Strict DGU* at a given day to be $vStrict DGU = 1\%$.
- **Rule 7:** $\forall SP_x$ who agree to *Install DGU* at t_i , ignore $\forall C_y$ where $SP_x \in C_y$ and C_y created before t_i while calculating $TLR_{avg}(SP_x)$ and $TST(SP_x)$. Remove $\forall G_y$ where $SP_x \in G_y$ and G_y created before t_i . Set $TG(SP_x) = postInstallDGURank$.
- **Rule 8:** $\forall SP_x$ who agrees to *Enable Strict DGU* at t_i , ignore $\forall C_y$ where $SP_x \in C_y$ and C_y created before t_i while calculating $TLR_{avg}(SP_x)$ and $TST(SP_x)$. Remove $\forall G_y$ where $SP_x \in G_y$ and G_y created before t_i . Set $TG(SP_x) = postEnableStrictDGURank$. Ignore SP_x presence in any future C_z in the calculations of the different Ranking Algorithms.

5.8 Threat Model

In Section 3.5, we argue that the current Trust Management Frameworks are doing excellent job in providing most of the Trust Requirements, except for the vital requirement of Continuous Data Control. For that, we introduce our Trust Framework Design in Section 4.1 where we borrow most of the components from existing Trust Frameworks like OpenID Connect and OAuth 2.0. Nevertheless, we introduce in that Section a completely new component we called the Auditor, which we explain in great detail in this Chapter, to provide Continuous Data Control.

Given that the Auditor is a completely new component that does not exist in the current literature the same way we design in this thesis, it is obligatory for us to study its threat model. That is because even the simplest forms of attacks against this Auditor, see the Simple Data Abusing Algorithms of Subsection 5.8.6, would not be tackled by currently available Security toolkit. Nevertheless, studying the threat models for the rest of the components appearing in Section 4.1 is out of our scope. For that, we are not going to consider, for example, the threat of Man in the Middle Attack since it is associated with the Encrypted Communication Channel component. Similarly, we will not consider the Social Engineering Attack aiming to convince the User or SP to voluntarily compromise their Credentials since it is targeting the humans using the Trust Framework rather than attacking a specific component within the Trust Framework.

In this threat model, the serious possible attacks to compromise the proposed Auditor are introduced. Some of these attacks are subjects of further simulation studies in Chapter 6 to better determine the extent of their influence on the overall Auditor performance. Other threats are not simulated for lack of resources issues. Nevertheless, Chapter 7 contains some theoretical mitigations and possible future research directions to tackle the presented threats as well as general discussions of their extent and possible effects in light of the simulation results and the proposed optimisations.

5.8.1 Users not Reporting All Abuse Cases

Ideally, all Users would report all the Abuse Cases they receive. However, this behaviour is unlikely to be true in real-life for a variety of reasons. Such reasons could be a cumbersome and confusing reporting procedure for the Users, lack of faith in the Trust Framework's ability to detect the malicious entities, or simply laziness. It is expected that the larger the number of ignored Cases by Users, C_{nI} , the greater the negative effect would be on the overall accuracy of the calculated Ranks. Particularly speaking, the TLR_{avg} Ranks would be the most sensitive metric to changes in C_{nI} . That is because of the fact that TL_{avg} calculations are dependent on the chronological order of appearance of each SP within the reported Case log and, hence, ignoring to report an Abuse could cause massive TL_{avg} distortion. It is also expected to negatively affect the TGT Ranks, with less severity, because the resulted potential colluding groups would get larger causing less certainty about whether they are colluding or not. Nevertheless, even if Users do not report all Cases, once an SP_x is considered malicious, it would be possible to reopen closed Cases were SP_x is involved and reconsider the possibility that SP_x is the actual Abuser rather than SP_y that was accused for this Abuse. This recalculation is vital for Auditors that are heavily dependent on the TLR_{avg} Ranking Algorithm. The simulation experiments in Chapter 6 confirm these expectations, see Section 7.2.1.

5.8.2 Malicious Users Reports

This is a tricky attack to detect. The goal of such an attack could be either to give MSPs high Trust Ranks or to give Trustworthy SPs lower Trust Ranks through reporting fake Cases. This attack affects mainly the TLR_{avg} Ranking Algorithm and $PIILs$ Record that is used as a selector for GT Agent Testing. This threat is not simulated in Chapter 6 but a potential mitigation is discussed in 7.3.7.

5.8.3 Making Money by Fooling the Testing Agents

This is an interesting possible attack against our proposed idea of Testing Agents presented in Subsections 5.3.2 and 5.4.1. Unlike the rest of the attacks, this attack aims to make money rather than to compromise Users' Credentials. When the Auditor has suspicions about a PSP_x , it would probably create one or more Testing Agents to interact with it. If the service provided by PSP_x is a paid service, then the Testing Agents would have to pay for the service they request to carry on their testing. Hence, it is possible for PSP_x to occasionally Abuse some of its Users to raise suspicions and let the Agents start their paid investigations. By employing some Malicious Users who would be colluding with PSP_x , PSP_x could guarantee that it would never be detected by the Auditor if it Abuses only those Malicious Users. That is, the Auditor would receive Abuse Cases regarding the actions of PSP_x only from the colluding Users. PSP_x may never take the risk of abusing any Uncolluding User. While we have not evaluated this threat in our Simulation Process, the proposed mitigation for Malicious Users in Subsection 7.3.7 should help tackling this type of attacks.

5.8.4 SPs Sharing Users' Credentials with MSPs Unintentionally

This threat is not an attack. Rather, it is a possible collaboration between an innocent SP_x with malicious SP_y without SP_x knowing the malicious intentions of SP_y . SP_x could simply decide to exchange Users' Credentials, which have Sticky Policies allowing such exchanges, with a popular SP_y for a variety of reasons like marketing. If it happens that SP_y is malicious, it would simply start abusing the Credentials it acquired through legitimate sharing with SP_x . Such accidental collaborations were assumed in the Simulation Model of Chapter 6 but without trying to anticipate the extent of their negative effects. However, some mitigations to minimise this threat effect are offered in 7.3.3.

5.8.5 Reverse Engineering the Auditor Algorithms by an ME

A Malicious Entity, ME, may inject the Trust Network with a huge number of malicious Users who would interact genuinely with MSPs belonging to the same ME. Such Users would report Cases to the Auditor and then would monitor how the TG values corresponding to those reported MSPs would be affected by the different Users reporting patterns. By doing this, the ME would be able to reconstruct a rough model of the Auditor's internal deployed Ranking Algorithms along with their associated Deployment Rules and, hence, would be able to optimize its Counter-Attacking Strategy accordingly.

Another method to gather reliable facts about the Auditor's internal Ranking Algorithms would be for a single person, or entity, to create several temporary SPs that would indulge in malicious activities and, once caught, it would capture the characteristics of the User that turns out to be the Testing Agent, i.e. its interaction pattern like frequency and variety. The constructed model would be fed to other SPs that belong to the same undercover owner in a feedback loop learning process. Every caught SP would contribute its Data to improve the reconstructed model and feed it to the learning feedback loop.

A more powerful method to infer the Auditor's Algorithms would be to alter the Attacking Settings and monitor whether these changes are affecting their published TG Rankings negatively, indicating that the Auditor is aware of their moves, or not. In the Uncolluding Attackers case, the main Ranking Algorithms that they should Counter-Attack are the TLR_{avg} and TST Ranking Algorithms, see Sections 5.2 and 5.3. The TLR_{avg} Ranking Algorithm is weak and easily fooled by moderate Attacking settings. Hence, a wise Auditor is better off deploying the TST Ranking Algorithm to minimise the threats of the Uncolluding Attackers. Nevertheless, the TST Ranking Algorithm comes with the high cost of required Testing Agents. For that, the Auditor may wish to turn off the TST Ranking Algorithm at some periods, assuming the Uncolluding Attackers won't notice this closure, to minimise the Agents cost. If the Uncolluding Attackers got able to notice, or guess, this period of turned off

TST Ranking Algorithm, they could try take advantage by relaxing their attacking settings, to Abuse more Users without being caught. When doing so, they would keep an eye to see whether their official *TG* ranks published by the Auditor are being reduced significantly. If they are, that would signal that the Auditor is turning on the *TST* Ranking Algorithm and it is time to deploy more complex Counter-Attacking Strategies.

Similarly for the Colluding Attackers, they could start their Attacks with, for example, Strong Colluding Attacks. Once the Auditor deploys a Strong Colluding oriented Defensive Strategy, the Attackers would know it by observing their *TG* Ranks falling down gradually. When the Attackers are sure that the Defensive Strategy is Strong Colluding oriented, they could switch to Weak Colluding Attacks. Once they observe their *TG* Ranks falling gradually, they would know that the Auditor has finally discovered their new technique and, hence, they should switch back to Strong Colluding Attacking Strategy or perhaps a mixture of the two Attacking Strategies. See Section 6.7 for more details about the Defensive Strategies and the Counter-Attacking Strategies.

5.8.6 Misleading the Auditor with Simple Data Abusing Algorithms

It is expected that smart MSPs would do their homework and read this open source proposal to understand the basic building blocks and how to fool them. The following are some of the important Simple Data Abusing Algorithms that could be deployed by them:

- **Abuse-Delay Attack:** knowing that the Auditor deploys the *GPD* Algorithm, an MSP_x could launch this attack where it simply sets a timer to delay abusing the User to fool the *GPD* Algorithm by giving time for other, possibly innocent SPs, to appear in the Case log after MSP_x . The longer the delay, the higher the number of other SPs that would appear in the chain causing the Rank of MSP_x to get even more better.

- **Abuse-Bombarding Attack:** To make things even worse, the User may have only interacted once with MSP_x but MSP_x could combine the Abuse-Delay attack with a Abuse-Bombarding Attack where it would delay abusing its victim User by a long enough period before bombarding him with consecutive Abuses. That would result in mistakenly giving MSP_x a high $TL_{avg}(SP_x)$ while others, probably PSPs, would get low $TL_{avg}(SP)$ values. That is because within the short interval between each consecutive Abuse, it is highly likely that the User would have dealt with a PSP that would in turn appear as a sole SP in the reported Cases for each of the bombarded Abuse Cases.
- **Credential Drop and User-Drop Attacks:** knowing that the Auditor could create some form of Testing Agents, MSP_x may decides to deploy a Credential-Drop Attack where it decides to neglect abusing a specific Credential belonging to $User_y$. A special case of this attack would be the User-Drop Attack where MSP_x would neglect all the Credentials belonging to $User_y$. The dropping could be for a temporary period or could be permanent. In case of temporary dropping, continuous interactions for a long enough period would enable the Testing Agent to eventually get positive results. However, in case of permanent dropping, if the Testing Agent happened to be an ignored User or if the main Testing Credential happened to be the ignored Credential, then the Agent would never get an Abuse from the tested MSP_x and, hence, would always return negative results. The Dropping Attacking Algorithm could be simply random or could be based on some guessed or implied facts, e.g. the ratio of $\frac{Testers}{Users}$ or the interaction patterns of genuine Users like frequency and variety, see Subsection 5.8.5.

5.8.7 Colluding $M(P)SPs$

In spirit of getting reliable facts about how the Auditor works, MSPs could decide to collude by sharing their Users' DBs. To simplify the analysis of this threat, we would always assume the presence of a Malicious Entity, ME, that would combine all the collected Data by the collation's members for further analysis and to conclude whether

the collation's members could go ahead abusing a given User or not. In practice, the ME could be simply one of the collation's members or the whole analysis could be done in a distributed fashion rather than centralised as presented in this thesis.

There are two main benefits for MPSPs to take the risk of colluding and sharing their Users' DBs. The first would be to serve the purpose of verifying whether a $User_x$ is a genuine User who has interacted with other SPs in a similar pattern to the modelled average User, see Subsection 5.8.5. if $User_x$ does not behave per the average User model, the tested MSP could launch the User-Drop Attack to fool the Auditor, see Subsection 5.8.6.

The second benefit for colluding MPSPs, out of their colluding, would be the fact that even in case they got inspected by a GT Agent that would be dealing in purpose with a group of SPs, which could include some or all of the collation members, a large enough collation group would lead to reduce the detection accuracy of the GT Agent, see Subsection 5.4.1. That could be achieved by applying a policy where they would never Abuse any User who have not dealt with a specific number of the collation's members. The larger that number is, the less useful the corresponding $TGT(SP)$ values would get. Such large number combined with utilising the Simple Data Abusing Algorithms described in 5.8.6 to get high TL_{avg} values for the collation's members could render the final $TG(SP)$ values useless. That would be especially true given that the ST Agents would always return negative results since the collation's members would never Abuse a User who have only dealt with one MPSP and, hence, the collation's members would always get high $TST(SP)$ Ranks.

In details, the two colluding parameters that an ME has to set are:

- $N_{toAbuse_C}$: the minimum number of colluding MPSPs a victim $User_x$ should deal with using $Credential_y$ before ME allows abusing this $Credential_y$.
- $N_{toAbuse_U}$: the minimum number of colluding MPSPs a victim $User_x$ should deal with using any of its Credentials before ME allows abusing $User_x$.

It should be noted that $N_{toAbuse_C}$ is a more conservative attack setting compared to $N_{toAbuse_U}$. That is because it is always true that $N_{toAbuse_C} \leq N_{toAbuse_U}$ since even if an ME set $N_{toAbuse_C} > N_{toAbuse_U}$, a victim $User_x$ who has dealt with $N_{toAbuse_C}$ colluding MPSPs using $Credential_y$ would automatically satisfy the condition of dealing with at least $N_{toAbuse_U} == N_{toAbuse_C}$. In other words, even if a victim $User_x$ has dealt with the $N_{toAbuse_U}$, the ME won't allow its collation's members to Abuse $Credential_y \in User_x$ unless he uses $Credential_y$ to interact with at least $N_{toAbuse_C}$. Setting $N_{toAbuse_C} > 1$ would guarantee avoiding the possibility of being caught by an ST Agent whose testing Credential is in fact $Credential_y$, see Subsection 5.3.2. Nevertheless, setting $N_{toAbuse_U}$ high and $N_{toAbuse_C}$ low combined with a long enough Abuse-Delay Attack, see Subsection 5.8.6, would lead to generating long *PILL* entries that could include only one or two colluding MPSPs and a lot of innocent PSPs, which would badly affect the accuracy of both the TGT_{sg} and TGT_{sc} Ranking Algorithms, see Subsections 5.4.2 and 5.4.3. If the collation includes only two or three MPSPs, it might be easy to detect. However, the attacks we are describing could include more than that. Those colluding MPSPs would have already dealt with other Credentials corresponding to a victim User and, hence, they would satisfy the condition of high $N_{toAbuse_U}$ without necessarily leaving footprints of all the collation members in all the reported Cases of a single Credential. The $TGT_{wc}(SP)$ Ranking Algorithm tries to tackle this issue by analysing all the PSPs a victim User has dealt with regardless of the used Credential. Nevertheless, this approach has its own shortcomings, see Subsection 5.4.4.

The colluding threat could have many variations, below are some important ones:

- **Colluding MPSPs:** in this simple colluding form, a group of MPSPs would be involved in the collation. Once the ME verifies that a given $User_x$ has dealt with at least the specified $N_{toAbuse}$ parameters, it would disseminate to all the collation MPSPs that it is alright to start abusing $User_x$ per their own applied Simple Data Abusing Algorithms, see Subsection 5.8.6. Of course, if a colluding MPSP decided to User-Drop $User_x$ at the time of acquiring his Credentials, it would not share those Credentials with the ME and, hence, this encounter won't be counted in the $N_{toAbuse}$ counters. This setting could significantly

reduce the number of Users a collation could Abuse. Nevertheless, it would reduce the collation's footprint in the Trust Network making it more challenging to detect its members by the Auditor.

- **Colluding MPSPs with a large pool of MSPs:** given it is expensive and risky to arrange a collation among a large number of MPSPs, this variation requires a smaller number of MPSPs to collude along with a large pool of MSPs which are inexpensive to create by the ME. The MPSPs would act completely genuine to prevent the risk of getting detected and then banned by the Auditor. Nevertheless, once an MSP interacts with a given User, it would double check with the ME whether that User has satisfied the $N_{toAbuse}$ conditions. If it does, it would get the green light to Abuse. Since MSPs have naturally very low footprint because of their unpopularity, they are very hard to detect specially if they are involved in a collation. However, this variation, despite its very low risk for the attackers, is not expected to give high gain for the attackers in terms of the volume of Abuse they could send.
- **Advanced MPSPs and MSPs Colluding to bypass DGU restrictions:** when the Auditor deploys DGU defensive rules and Strategies, see Subsection 5.6.3 and Section 5.7, this advanced variation could be utilized by the ME to counter these defences. It works in a similar manner to the Colluding MPSPs variation but once an $MPSP_x$ got detected by the Auditor and gets asked to *Install DGU*, it would bypass this restriction by legally sharing its Users' DB with a random artificial MSP_y that the ME could create for this purpose, see Subsection 5.8.4. MSP_y would then send the shared DB to the ME on behalf of $MPSP_x$ so the $N_{toAbuse}$ counters would be updated as if $MPSP_x$ has not *Installed DGU*. If $MPSP_x$ needs to Abuse a given User, it would simply ask MSP_y to do it in its behalf. Since it is possible to create a very large pool of artificial MSPs by the ME, it is possible for $MPSP_x$ to switch the MSP it shares with its DB to prevent the Auditor from observing any suspicious ties between $MPSP_x$ and MSP_y leading to banning both of them.

5.8.8 *MPSP Committing Suicide Attack*

If an $MPSP_x$ got caught after a long period of time, the amount of ignored Cases and recalculated Ranks as a result of banning $MPSP_x$ and accusing it for most of the prior Cases instead of the other SPs who were found guilty, see Section 5.7, could severely affect the integrity of the Auditor. As a result, the Auditor could halt for a period of recovery time $\tau_{MPSP-Halt}$. The recovery speed would rely upon new Users joining the Trust Network or existing Users generating new Credentials to interact with the available SPs in the Network.

Knowing this fact, an ME may try to launch a Suicide Attack by letting one of its MPSPs act genuinely for a long period of time and, then, it would alter its behaviour to start acting maliciously without any efforts to hide its identity to get caught. Once it gets caught, the Auditor would get into the recovery period, $\tau_{MPSP-Halt}$, which would be the perfect timing for other $M(P)SPs \in ME$ to Abuse all the Users already in their DBs without their Ranks being affected immediately. In theory, the recovery period could be calculated as shown in Equation 5.11. It should be noted that to make any sense of that Equation, the Auditor should maintain accurate statistics and approximations to get the current values for the Equation's variables, see Subsection 5.6.4.

$$\tau_{MPSP-Halt} = \frac{C_{MR}*(SP_{nMSP-est-new}-SP_{nMSP-est-old})}{(C_{GR}*U_{GR})+(C_{GR-newPII}*U_n)-(C_{MR}*SP_{GR-M})}$$

where $\tau_{MPSP-Halt}$ is the recovery period after detecting an MPSP and $\tau_{MPSP-Halt} \in \mathbb{Z}_{\geq 0}$,

C_{MR} is the minimum # of Cases to recover per SP and $C_{MR} \geq 1$,

$SP_{nMSP-est-new}$ is the estimated # of M(P)SPs after the Suicide Attack and $SP_{nMSP-est-new} \geq 1$,

$SP_{nMSP-est-old}$ is the estimated # of M(P)SPs before the Suicide Attack and $SP_{nMSP-est-old} \geq 1$,

C_{GR} is the reported Cases Growth Rate per User and $C_{GR} \geq 0$,

U_{GR} is the Users Growth Rate and $U_{GR} \geq 0$,

$C_{GR-newPII}$ is the reported cases Growth Rate for newly generated PII Data per User

and $C_{GR-newPII} \geq 0$ and $C_{GR-newPII} \subset C_{GR}$,

U_n is the current # of Users and $U_n \geq 1$,

SP_{GR-M} is the MSPs Growth Rate and $SP_{GR-M} \geq 0$.

(5.11)

Equation 5.11: Recovery Period after Detecting an MPSP, $\tau_{MPSP-Halt}$

In Equation 5.11, it should be noted that: $\tau_{MPSP-Halt} \in \mathbb{Z}_{\geq 0}$ because any negative time would mean the recovery happening in the past, which does not make sense in real-life. In fact, when the fraction denominator gets much larger than the numerator, the recovery time would get much shorter and vice versa. It should also be noted that while the growth rates could be close to 0 when saturation is reached, the absolute numbers of current SPs and Users cannot be < 0 . In addition, $C_{MR} \geq 1$ because without at least one reported case of Abuse, the Auditor won't be triggered to investigate. Another point to note is that $C_{GR-newPII} \subset C_{GR}$. Finally, it is worth mentioning that it should be safe to replace Equation 5.11 with Equation 7.1 as shown in subsection 7.2.7.

When the $\tau_{MPSP-Halt}$ value approaches a negative value or ∞ , then it should be assumed that there is almost no hope to recover. That is, the Auditor would get into irrecoverable halt when the fraction numerator approaches ∞ or the denominator gets

≤ 0 . Based on these facts, the following scenarios has the power to lead the Auditor towards this point of irrecoverable halt:

- When C_{MR} approaches ∞ , i.e. when the Auditor cannot detect an M(P)SP regardless of how large is the number of reported Cases. That is in fact more like an internal problem that could be controlled by improving the deployed Ranking Algorithms. However, it would be challenging to reduce it significantly without introducing radical changes to the proposed Algorithms in this research.
- When $(SP_{nMSP-est-new} - SP_{nMSP-est-old})$ approaches ∞ , see Subsection 5.8.9.
- When C_{GR} and eventually its subset $C_{GR-newPII}$ approaches 0. That would be the case when the Trust Network saturates and/or when Users lose faith on the Network and stop interacting with SPs within it.

It is interesting to note that when the Network starts with a high C_{MR} value, that would lead to Users losing faith in the system and, hence, declining values of the following set of variables $\{C_{GR}, U_{GR}, U_n\}$, which would make the $\tau_{MPSP-Halt}$. In addition, the more the Network gets matured and saturated, the smaller the values of $\{C_{GR}, U_{GR}\}$ would get. In other words, unless $U_n \gg SP_{nM}$, C_{MR} should get smaller as the time pass. Otherwise, the $\tau_{MPSP-Halt}$ would continue to get larger until it reaches the point of irrecoverable halt.

5.8.9 MSPs Registration Bombarding

When a Suicide Attack takes place, the ME launching such attack should be expected to be managing a more holistic battle behind the scenes, i.e. this attack would be probably followed by other Targeted Attacks to defeat the Trust Network defensive lines, see Attacks 2.0 in Section 2.4. Hence, it should not be a surprise if the ME would follow the Suicide Attack directly with an MSPs Registration Bombarding Attack where almost all of the new SPs registering in the Network would be malicious. In other words, the Equation of $\tau_{PSL-Halt}$ would approximate to Equation 5.12.

$$\tau_{MPSP-Halt} = \frac{C_{MR} * SP_n}{(C_{GR} * U_{GR}) + (C_{GR-newPII} * U_n) - (C_{MR} * SP_{GR})} \quad (5.12)$$

where SP_n is the current number of SPs and $SP_n \geq 1$,

SP_{GR} is the SPs Growth Rate and $SP_{GR} \geq 0$.

**Equation 5.12: Recovery Period after Detecting MPSP assuming the
MSPs Registration Bombarding Attack, $\tau_{MPSP-Halt}$**

Per Equation 5.12, in Networks, specially saturated ones, where $SP_{GR} \gg U_{GR}$, SP_n would eventually gets $\gg U_n$ leading to $\tau_{MPSP-Halt} \approx -\infty$, i.e. reaching the irrecoverably point. The Auditor may think about blocking new SPs from joining the Network during the estimated recovery period to mitigate this scenario. However, this would be ineffective if the ME has already bombarded the Network with MSPs prior to launching the Suicide Attack.

Chapter 6. Trust Auditor Model, Simulation and Results

In Chapter 5, we have introduced the basic conceptual building blocks for an effective governing Auditor to be the core component of our proposed Continuous Trust Framework Design as presented in Chapter 4. In this Chapter, a Simulation Model is introduced in Section 6.3 to analyse how such an Auditor would behave in real-life given different sets of Users and SPs with a population of Malicious Entities, MEs, deploying different Trust attacks per the threat model presented in Section 5.8. The Simulation Process of this Chapter allows us to get a general picture of how our Auditor could perform in Trust Networks consisting of thousands or even millions of nodes. Through simulation, it is possible to let a large enough number of nodes to interact with that Auditor for a fairly long period of time to capture the picture of its performance. Real life development of such large number of nodes and getting volunteers to manage them by adopting different behaviours according to the testing goals for a long period of time would be impractical and unnecessary at this stage of the research.

To initiate the Simulation Process, we developed a Java Simulation Model, see Section 6.3. That model basically implements the basic Auditor building blocks described in Chapter 5 and assumes the Users of the Trust Network would build their perceived Trust based on the Trust Model we present in Section 6.4. Further, our Simulation Process utilizes a cycle-driven simulation library called PeerSim to run our Simulation Model and capture the results, see Section 6.2. To correctly comprehend the effects of the various Defensive factors and the Counter-Attacking settings, we defined the settings we are interested in testing and, then, designed sequential partial factorial sets of experiments grouped as shown in Section 6.5. These sequential experiments are grouped in eight Experimental Stages and their results are described in Section 6.6.

While conducting the Experimental Stages, we realised some defensive patterns that are countered by other patterns of attacks. By the end of the Experimental Stages, we were able to classify three unique Defensive Strategies as well as several optimized settings, Flavours, for each of those Defensive Strategies along with the two main Counter-Attacking Strategies as we elaborate in Section 6.7.

During the Simulation Process, we optimized the Defensive Strategies by testing altering their settings under different Environmental Settings, i.e. genuine *Normal Nodes* populations, *Malicious Nodes* populations, and the level of *Users' Ignorance Rate* in reporting the received Abuse Cases. We noticed some variations in the performance under different variations of the Environmental Settings. That led us to further investigate the matter and, hence, came up with the Malicious Density Theory that we present in Section 6.8. In that theory, we show that the *MPSPs Detection Rates* would differ depending on the Malicious Nodes Density within the Trust Network.

At the end of the Experimental Stages, we tested all the Defensive Strategies Optimized Settings against each other as well as the case of deploying nothing. Section 6.9 shows the detailed results of this comparison. In that section, it is clear that our proposed Strategies would significantly improve the current *Users Trust Rate* and reduce the *Compromised Credentials Rates*. That is of course not by completely preventing all possible attacks. Rather, it is by raising the rules of the game between the Defensive Auditor and the Counter-Attackers. That is, we make it very hard and expensive for an Attacker to compromise a Credential and, hence, we reduce the level of the overall abusing. Since this work is far from perfect, it has its limitations. A list of the limitations and mistakes we are aware of is presented and discussed in Section 6.10.

6.1 Simulation Scope

The Aim of this Simulation Process is: to **design** unique **Defensive Strategies** for the Auditor and to **optimize** their settings against different sets of colluding and Uncolluding M(P)SPs who would dynamically optimize their **Attacking Strategies**.

by generating a set of unique **Defensive Strategies' Flavours** Auditor's **Defensive Strategies**

Defensive Strategy refers to: a distinctive perceived behaviour of the Auditor's defensive settings. That is, the Defensive Strategy is a textual description of the overall effect the internal settings of the Auditor would cause in terms of detecting and banning suspicious SPs.

Defensive Strategy Flavour refers to: a unique optimized combination of defensive building blocks with a predefined set of weights and parameters as well as Deployment Rules, see Sections 5.1 and 5.7 for more details. A Defensive Strategy could have a large number of Flavours depending on the optimisation or manual settings by the Auditor.

Attacking Strategy refers to: a unique combination of malicious attacks and predefined attacking parameters, see Subsections 5.8.6 and 5.8.7 for more details.

Optimisation refers to: maximising certain features as well as minimising, or at least neutralising, unwanted features. Particularly, the optimisation process in the context of the Defending Auditor aims to:

- Maximise the **Perceived Trust** by the Network Users
- Minimise the **Number of Testing Agents** because of their operational costs (creation, engaging in costly interactions, and maintaining) and because a Network that is dominated by a population of Testing Agents is not worth joining by either SPs or Users
- Minimise the **Number of Undetected M(P)SPs**
- Minimise the **Number of Open Cases**
- Minimise the **Number of Compromised Credentials**
- Minimise the **Number of Banned Innocent (P)SPS**

- Minimise the **Number of Users Installing or Enabling Strict DGU**, to ease the bootstrapping process

On the other hand, the optimisation process in the context of the Attacking M(P)SP or ME aims to:

- Maximise the **Number of Undetected M(P)SPs**
- Maximise the **Number of Compromised Credentials**

It should be noted that due to resources limitations, not all of the threats discussed in Section 5.8 were considered in this simulation process. The potential effects of those omitted threats are discussed in Section 7.2. The threats and attacks that are within the simulation scope are:

- Users not reporting all Abuse Cases, see Subsection 5.8.1
- Innocent SPs sharing Users' Credentials with M(P)SPs unintentionally, see Subsection 5.8.4
- Misleading the Auditor with Simple Data Abusing Algorithms, see 5.8.6
- Colluding M(P)SPs, see Subsection 5.8.7

6.2 Development Environment

In developing and executing this simulation project, many different hardware, software, and libraries were utilized. The main components are:

- **Hardware:** A Dell Desktop Computer running Intel(R) Core (TM) i7 CPU 860 @ 2.80 GHz with 4030 MB of RAM memory
- **OS:** Microsoft Windows 7 Enterprise

- **IDE:** Eclipse Java EE IDE for Web Developers - Version: Luna Service Release 2 (4.4.2)
- **Programming Language:** Java - JVM 1.8.0
- **Simulation Library:** PeerSim 1.0.5 [122]
- **Statistics Software:** Minitab 17 with aid of MS Excel Sheets - Version: Professional Plus 2013

The code of this Simulation Model is published as an open-source project and can be found on: https://gitlab.com/Continuous_Trust_Simulator/Simulator.

6.3 Simulation Model

As stated in Section 6.2, the chosen library to simulate the proposed Auditor building blocks of Chapter 5 is PeerSim, for its simplicity and ease of use [122]. This library offers two simulation options: cycle-driven and event-driven [123]. While the event-driven option would be more realistic, the cycle-driven is less challenging to execute. The main disadvantages of the cycle-driven approach are the lack of the transport layer and concurrency simulation. However, Network latency and communication efficiency simulations are not in this project scope, see Section 6.1, and not necessary at this stage of the research and, hence, the cycle-driven approach has been chosen to develop the simulation model. It should be noted that in this approach, each execution cycle represents a unit of time that could be anything from a millisecond to a full year or even more. The interpretation of that unit of time depends on the context of the simulation and how the model is designed to act in each different cycle. In the context of this research project, we choose to set this unit of time to be a single day. That may not be accurate enough to represent real Users' interactions. However, this limitation has little impact on the validity of this simulation as discussed in Section 6.10.

Per [123], the main components of a PeerSim simulation model are as following:

- **Network Nodes:** Main interacting nodes in the Network.

- **Initializers:** Responsible for initializing new instances of Network nodes.
- **Controllers:** Execute the logic of the Simulation at different running cycles by triggering the Network Nodes to behave in certain ways per the experiment settings.
- **Observers:** Capture metrics of interest, in the form of customised Data files or objects for further analysis.
- **Configuration Files** Set the experiment settings and the values of the simulated factors.

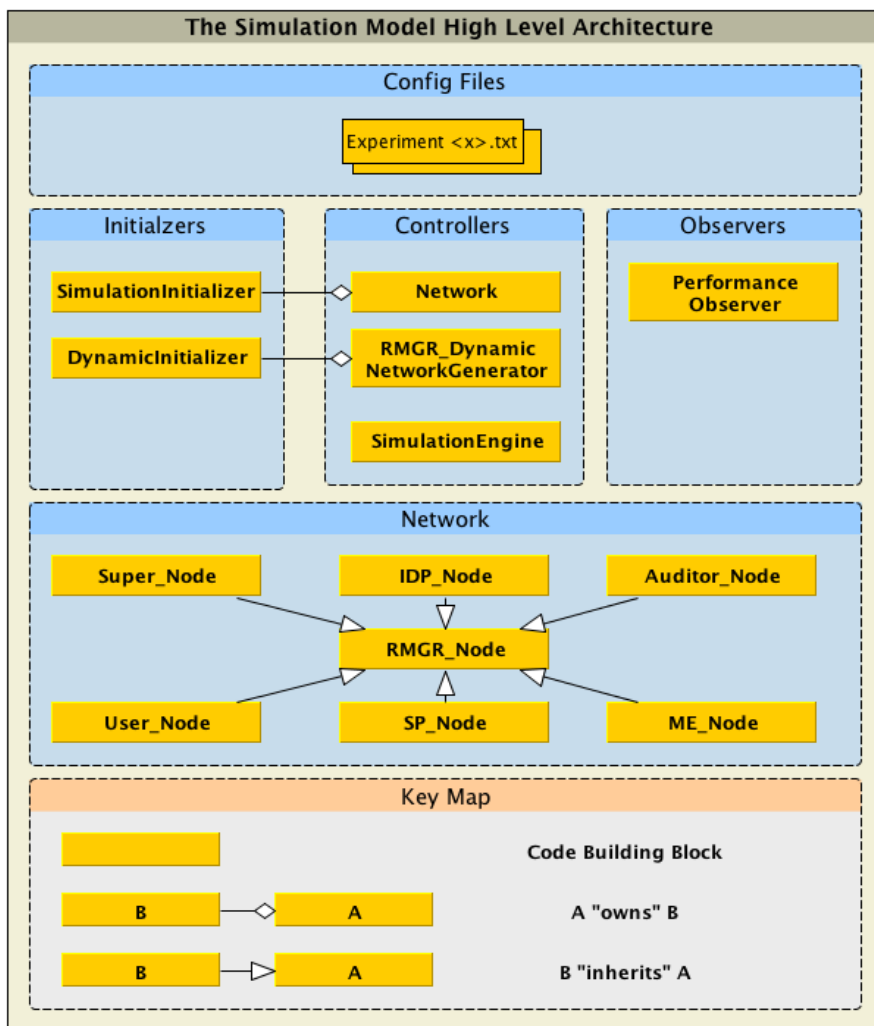


FIGURE 6.1: The Simulation Model High-Level Architecture

Our Simulation Model is designed to analyse the behaviour of the real Network residents without actually implementing them. That is, unlike the presented prototype in Section 4.2 where real-life Users, SPs, IDP, and an Auditor were deployed exchanging real Data, the nodes in this Simulation Model are Data objects that do not exchange Data the same as real-life nodes would do. For instance, in real-life, when a User wishes to get a service from an SP, it would ask the IDP to forward its Credentials to that SP and, in turn, that SP would directly contact that User. However, in this Simulation Model, the IDP would forward the User's Credentials to the SP and record in its logs that the SP has accessed the User's Credentials without any real Data exchange between the User and the SP like in the real-life scenario.

Figure 6.1 illustrates a high level class diagram of our Simulation Model as realised by the PeerSim convention. Note that the details of the internal methods, variables, and utility classes are omitted in this Figure. While the whole code along with its detailed documentation is available as an open source project, we will give a brief introduction to the listed general components in Figure 6.1.

Config. Files:

- **Experiment** $\langle x \rangle$.**txt:** Each of these Configuration files includes the factors' settings, the number of simulation cycles, the number of times the experiment should be repeated, and the PeerSim meta instruction to link the classes.

Initializers:

- **SimulationInitializer:** A helper object that initializes the first Nodes that bootstrap the Network.
- **DynamicInitializer:** A helper object that initializes the Nodes that dynamically join the Network.

Controllers:

- **Network:** Starts at the beginning of each experiment to create the required number of initial Nodes as specified in the Configuration File.
- **RMGR_DynamicNetworkGenerator:** Dynamically adds more Nodes at the beginning of each execution cycle per the Growth Rates sets in the Configuration File and the required new Testing Agents by the Auditor.
- **Simulation Engine:** Manages how to execute the new execution cycles with triggers to the different Network Nodes to perform different tasks as required by the simulation setting. For example, the Auditor would be required to update the *TGR* Ranks and send them back to the IDP at the set frequency of cycles in the Configuration File.

Observers:

- **PerformanceObserver:** Activated at the last execution cycle to gather and analyse the statistics collected by the Network Nodes and, then, publish its summary in the form of textual Data Files for further analysis using Minitab software, see Subsection 6.5.3 for more details.

Network Nodes:

- **RMGR_Node:** Contains the basic identifiers and methods needed by all the nodes such as the random number generators. Those generators are needed by all the different nodes to make dynamic decisions based on pre-set probabilities in the Configuration Files to mimic the desired behaviour of the simulation experiment.
- **Super_Node:** Acts like an internal Data holder that is needed to ease the Simulation Process. For example, when the Auditor wishes to dynamically create new Testing Agents, it would pass their details to the Super_Node that would, in turn, makes it readable for the RMGR_DynamicNetworkGenerator in the following execution cycle.

- **IDP_Node:** Resembles the Identity Provider that stores the Users' Credentials and pass them to the approved SPs by their owners. The IDP would always store access logs and transfer them to the Auditor whenever requested to do so in case of reported Abuse Cases by the corresponding User, see Section 4.2.
- **Auditor_Node:** Implements all the building blocks of Section 5.1 that are controlled by some Deployment Rules, see Section 5.7.
- **User_Node:** Resembles a real-life User that interacts in each cycle by a dynamic probability depending on its perceived Trust of the Network. This User ignores reporting a received Abuse at a pre-set probability and would also increase his overall Trust or decrease it depending on how effective the Auditor is at resolving the currently reported Abuse Cases as well as how most of the current Users Trust the Network, see 6.4.
- **SP_Node:** Has different versions: (P)SP, Uncolluding M(P)SP, and Colluding M(P)SP. The innocent (P)SP version do nothing but genuinely fulfilling the request of the User. The Uncolluding M(P)SP deploys some Simple Data Abusing Algorithms, see Subsection 5.8.6, with pre-set settings in the Configuration Files per the simulated behaviour. The Colluding M(P)SP would also deploy similar Simple Abusing Algorithms but would, in addition, pass its acquired Credentials to the ME_Node that it is associated with so that the ME_Node could decide whether a certain Credential or User could be attacked per its Colluding Attack settings that are controlled as well in the simulation settings, see Subsection 5.8.7.
- **ME_Node:** An entity managing a group of M(P)SPs, see Subsection 5.8.7.

6.4 Trust Model

For the sake of measuring the effectiveness of our introduced Auditor in the simulation experiments, we needed to implement a Trust Model that simulates the Users' attitudes when it comes to Trusting online SPs and their reactions when they receive

Abuse Cases. The Technology Acceptance Model, TAM, is a theory introduced by Davis in 1989 to model the Users' attitudes toward using technology based on the concerned technology perceived usefulness and ease of use [124]. Various extensions to that theory came on the following decades to incorporate the concepts of perceived Security, Privacy, and Trust as seen in [125] and [126]. Nevertheless, we have opted not to rely on this theory for various reasons. First, while this theory incorporates various elements in deducing the Users' Trust such as perceived usefulness and easiness, our simulation experiment is concerned with studying the effects of a set of measures that would improve only one aspect: Privacy. For that, implementing this generic model would add an extra layer of complication to our small project that we could not afford. Second, most of the conducted studies to generate the TAM models are based on groups of students rather than the general public and, hence, there is a potential bias on the generated models. Given the model's inaccuracy, due to the potential bias, and implementation complexity, we concluded that it is not justified to invest in incorporating these models in our simulation study. Instead, we opted for a minimal arbitrary model to get started with the intention to refine our Trust Model in the future experiments if it is needed. Fortunately, we found that our Trust Model was fair enough and there is no need to refine it. That is because our introduced set of defences raises the bar of the game between the defending Auditor and the Counter-Attackers to the point that the generated attacks would be so minor that it would not affect the overall perceived Trust, see Subsection 6.10.1 for more details.

$$trust_t = trust_{t-1} + (trust_{t-1} * \frac{IncreaseRate}{100}) \quad (6.1)$$

: where $0 \leq trust_t \leq 100$ and the default values are:

$$trust_0 == 50, IncreaseRate == 1.$$

Equation 6.1: Trust increase in each cycle without an open Case

$$trust_t = trust_{t-1} - (trust_{t-1} * \frac{DropPostAbuse}{100})$$

DecreasePeriodPostAbuseCount =

$$DecreasePeriodPostAbuseCount + DecreasePeriodPostAbuse \quad (6.2)$$

: where $0 \leq trust_t \leq 100$ and the default values are:

$trust_0 == 50$, $DropPostAbuse == 10$, $DecreasePeriodPostAbuse == 25$.

Equation 6.2: Trust decrease after receiving an Abuse Case

(if(DecreasePeriodPostAbuseCount > 0)) :

$$trust_t = trust_{t-1} - (trust_{t-1} * \frac{DecreaseRate}{100})$$

$$DecreasePeriodPostAbuseCount - - \quad (6.3)$$

: where $0 \leq trust_t \leq 100$ and the default values are:

$trust_0 == 50$, $DecreaseRate == 1$.

Equation 6.3: Trust decrease in each cycle with open Cases

$$\begin{aligned}
trust_t &= trust_{t-1} + (trust_{t-1} * \frac{IncreasePostAbuseResolved}{100}) \\
DecreasePeriodPostAbuseCount &= DecreasePeriodPostAbuseCount - \\
&(DecreasePeriodPostAbuse - PeriodToClose)
\end{aligned} \tag{6.4}$$

: where $0 \leq trust_t \leq 100$, $PeriodToClose \geq 0$

and the default values are:

$trust_0 == 50$, $IncreasePostAbuseResolved == 5$.

Equation 6.4: Trust increase after resolving Abuse Case

In our Trust Model, we assumed that the initial Trust at the bootstrapping phase is 50% in an attempt to model the real-life state where Users would have some unproven doubts toward dealing with any new entrant to the market. Of course, the percentage of doubt would vary from person to person based on their previous experiences and the way each individual decision making process. Nevertheless, we needed a simple arbitrary point to start with and, hence, we selected the 50% middle point. At each new execution cycle, if the User did not receive any Abuse Case, or if it received but ignored, the User's Trust on that Trust Framework is expected to grow, but at a slow pace because we know that Trust is "hard to build and easy to lose [16]". For that, we choose to grow that Trust by 1% at each cycle without any received Abuse Case until reaching the maximum of 100% as listed in Equation 6.1.

When an Abuse Case is received, many real-life Users are expected to simply ignore it due to laziness or unimportance and, hence, that won't affect the overall Trust. However, when a User decides to report an Abuse Case, that would imply that this Abuse had damaged to some extent their overall Trust in the system. Hence, the Trust is expected to drop dramatically, again because we know that Trust is "hard to build and easy to lose [16]". For that, we choose to drop that Trust by 10% following an Abuse Case as listed in Equation 6.2. Afterwards, we expect most Users who opted to

be part of our newly invented concept of Continuous Trust Management Framework would be waiting anxiously for a response from the Auditor to confirm detecting the Abuser. The more time it takes to get that response, the more of the User's Trust on the Framework would be lost. For that, we choose to drop that Trust by a rate of 1% at each following execution cycle for a total of 25 cycles, for each reported Abuse Case, as listed in Equation 6.3. The reason for the arbitrary point of 25 cycles before stopping the Trust decrease due to one Abuse incident is the fact that human Users tend to forget as time passes and continue dealing with an SP that is useful and easy to use despite the Privacy issues that it may have [125].

The Trust decrease due to a reported Abuse Case is expected to stop once the Auditor detects the guilty SP, as a way of appreciation to the sincere efforts of the Auditor at providing the Continuous Data Control Attribute that is lacked in most other Trust Framework. Of course, the Trust increase is not expected to be as much as the initial Trust decrease that followed the reported Abuse Case because Trust is "hard to build and easy to lose [16]". For that, we choose to increase that Trust by 5% as listed in Equation 6.4.

$$newCred_t = randBoolean(1 - (\frac{NewCredRate}{100} * \frac{trust_t}{100})) \quad (6.5)$$

: where $randBoolean(x)$ is a random generator returning false at probability x and the default value is: $NewCredRate == 10$.

Equation 6.5: Probability of generating a Credential at t

$(If(newCred_t)) :$

$$strictCred_t = randBoolean(1 - \frac{StrictCredProb}{100}) \quad (6.6)$$

: where $randBoolean(x)$ is a random generator returning false at probability x and the default value is: $StrictCredProb == 10$.

Equation 6.6: Probability of assigning a Strict Sharing Policy to a new Credential at t

$(If(!strictCred_t)) :$

$$shareWithTop_t = randBoolean(\frac{trust_t}{100}) \quad (6.7)$$

: where $randBoolean(x)$ is a random generator returning false at probability x

Equation 6.7: Probability of assigning a Share with Any Policy to a new Credential at t

When it comes to generating Credentials, email addresses or credit card numbers for example, we expect that the more Trust the User has in the Trust Framework the more Credentials she will be willing to use in that Framework. For that, all the Credentials generation Equations in our Trust Model depends on the User's overall perceived Trust. In that Trust Model, we set an arbitrary maximum probability of generating a new Credential by a given User at a given simulation cycle to be 10%. This probability could be further reduced by a negative linear relation with the User's Trust as listed in Equation 6.5. In addition, all new Credentials could be generated

with one of three Sticky Policies, see Subsection 4.4.1 for more details about the Sticky Policies:

- **strict**: this prevents the SP acquiring the Credential from sharing it with any other SP.
- **shareWithTop**: this enables the SP acquiring the Credential to share it with only PSP providers because they are thought to be more genuine and less likely to be malicious.
- **shareWithAny**: this enables the SP acquiring the Credential to share it with any SP in the Network.

In this Trust Model, it is assumed that 10% of the new Credentials created by a given User would be strict, representing valuable and important Credentials like credit cards details or health records, as listed in Equation 6.6. The remaining 90% of Credentials would be dynamically adjusted based on the current level of Trust. That is, the less Trust in the Network, the more Credentials would be assigned shareWithTop Sticky Policies. Any Credential that does not get the shareWithTop policy would automatically be assigned a shareWithAny policy as listed in Equation 6.7.

6.5 Simulation Approach

The following are the main steps illustrating the simulation approach we have applied to study and optimize our proposed Auditor of Chapter 5:

- Defining the factors of interest and the interesting outputs for further optimisation, see Subsection 6.5.1.
- Defining the main Experimental Stages, see Section 6.6.
- Designing mini-experiments for each Stage based on the partial factorial principle described in Subsection 6.5.2.

- Executing the mini-experiments sequentially. That is, we fed the results of one experiment to the following experiment in the list. We analysed the raw output results statistically using Minitab software along with MS Excel sheets to obtain the new optimized values for the following mini-experiment in the list, see Subsection 6.5.3.

6.5.1 Factors of Interest

Experiment Settings				
	Factor	Low	High	Notes
	Replicas	var		Depends on total of factors - 30 when all factors are constant and goes up to 3840 when analysing the Max. of 11 factors

TABLE 6.1: Final List of Factors of Experiment Settings

Environmental Factors				
	Factor	Low	High	Notes
Grouped	iniUsers	50	130	ini prefix: initial number of nodes. GR suffix: percentage growth rate of nodes per simulation cycle.
	iniPSPs	8	20	
	iniSPs	16	40	
	UsersGR	0.05	0.2	
	SPsGR	0.05	0.1	
	PSPsGR	0.05	0.1	
Grouped	iniMPSPs	4	10	ini prefix: initial number of nodes. GR suffix: percentage growth rate of nodes per simulation cycle.
	iniMSPs	8	20	
	iniME_MPSPs	5	7	
	iniME_MSPs	8	12	
	MPSPsGR	0.05	0.1	
	MSPsGR	0.05	0.1	
	ME_MPSPsGR	0.05	0.1	
	ME_MSPsGR	0.05	0.1	

TABLE 6.2: Final List of Environmental Factors

Users' Controlled Factors. See Section 6.4 for more details.				
Factor	Low	High	Notes	
Ignorance Rate	20	80	Probability a User would ignore reporting an Abuse Case	

TABLE 6.3: Final List of Users' Controlled Factors

SPs' Controlled Factors. See Subsection 5.6.3 for more details.			
Factor	Low	High	Notes
Accept DGU Prob	0	40	Probability an SP voluntarily accepts to <i>Install DGU</i> , if offered to by Auditor.
Accept Strict DGU Prob	0	20	Probability an SP voluntarily accepts to <i>Enable Strict DGU</i> , if offered to by Auditor.

TABLE 6.4: Final List of SPs' Controlled Factors

Auditor's and IDP's Controlled Factors				
	Factor	Low	High	Notes
	TG Ranking Settings. See Section 5.5 and Subsection 5.4.5 for more details.			
	$W_{TLR_{avg}}$	20	80	"TLRavg Weight" in Conf. Files.
	W_{TST}	20	80	"TST Weight" in Conf. Files
	W_{TGT}	20	80	"TGT Whole Weight" in Conf. Files.
	W_{sg}	20	80	"TGT Weight" in Conf. Files.
Grouped	W_{sc}	10	40	"TGT Colluding Weight" and
	W_{wc}	40	10	"TGT Weak Colluding Weight" in Conf. Files.
	TLR_{avg} Settings. See Section 5.2 for more details.			
	Suspicious SP Rank	20	40	
	Suspicious SP Banning Rank	5	15	
Grouped	Sufficient TLRus PSP	5	20	Min TLRu ranks to ban (P)SP based on its TIR_{avg}
	Sufficient TLRus SP	2	10	
	TST Settings. See Sections 5.3 and 5.7 for more details.			
	DeployST	false	true	Turning on/off all the ST settings
	$\tau_{ST-maxLife}$	20	80	"ST Max Life Time" in Conf. Files.
Grouped with GT Report Abuse	ST Report Abuse	false	true	Reporting Abuse like ordinary Users (using Unimportant Credentials), "ST Report Spam" in Conf. Files.
	PSL ST Selector	false	true	ST Selectors. See TST Deployment Rules in 5.7.
	Suspicious Nodes ST Selector			
	Top TLR_{avg} ST Selector			
	Bottom TLR_{avg} ST Selector			
	IIR ST Selector			
	TGT Settings. See Sections 5.4 and 5.7 for more details.			
	DeployGT	false	true	Turning on/off all the GT settings

Auditor's and IDP's Controlled Factors				
	Factor	Low	High	Notes
	$\tau_{GT-maxLife}$	20	80	"GT Max Life Time" in Conf. Files.
Grouped with ST Report Abuse	GT Report Abuse	false	true	Reporting Abuse like ordinary Users (using Unimportant Credentials), "GT Report Spam" in Conf. Files.
	PCR Threshold	2	7	
	PCR Weak Threshold	2	7	
	Ignore Old G Ranks	false	true	TGT ignores all Gs containing a banned SP
	GT Max SP Num	2	7	Max unpopular SPs in each G
	GT Max Size	2	7	Max G size
	PIIL GT Selector	false	true	GT Selectors. See TGT Deployment Rules in 5.7.
	PSL GT Selector			
	Suspicious Nodes GT Selector			
DGU Settings. See Subsection 5.6.3 for more details.				
	DeployDGU	false	true	Turning on/off all the DGU settings
	Post DGU Installation Rank	20	80	
Grouped	Offer DGU Prob	0	20	
	Offer Strict DGU Prob	0	10	

TABLE 6.5: Final List of Auditor's and IDP's Controlled Factors

Attackers' Controlled Factors. See Subsections 5.8.6 and 5.8.7 for more details.				
	Factor	Low	High	Notes
Begin: Factors to be duplicated for M(P)SPs \in MEs				
	Abuse Delay Period	5	50	Simple Data Abusing Algorithms' factors, see Subsection 5.8.6. "Spam" instead of "Abuse" in Conf. Files
	Abuse Bombarding Period	5	50	
	Abuse Drop User Rate	20	80	
	Abuse Drop Credential Rate			
End: Factors to be duplicated for M(P)SPs \in MEs				
	$N_{toAbuse_U}$	1	5	"ME N MIN MPSPs to Spam User" and "ME N MIN MPSPs to Spam Credential" in Conf. Files.
	$N_{toAbuse_C}$			

TABLE 6.6: Final List of Attackers' Controlled Factors

Monitored Outputs. See Section 6.1 for more details.	
Output	Notes
All Outputs ending with <> are duplicated four times to record the same output during 4 different cycles: 125, 250, 375, and 500	
Trust In Network Users	Avg of all Users Trust by the end of simulation
Active Agents to Users at <>	PCT of Agents among total Users
PCT Undetected MPSPs at <>	
PCT Undetected MSPs at <>	
PCT Undetected Popular MPSPs at <>	MPSPs ∈ ME colluding normally
PCT Undetected Unpopular MPSPs at <>	MPSPs ∈ ME colluding with a large pool of MSPs
Open cases at <>	
PCT Compromised Share with Top Credentials at <>	
PCT Compromised Share with Any Credentials at <>	
PCT Abuse by MPSPs at <>	“Spam” instead of “Abuse” in the raw results.
PCT Abuse by MSPs at <>	
PCT Abuse by Popular Colluding ME at <>	
PCT Abuse by Unpopular Colluding ME at <>	
PCT PSPs Banned Guilty at <>	PCT of PSPs banned because they were thought Guilty
PCT PSPs Banned DGU at <>	PCT of PSPs banned for refusing to <i>Install DGU</i>
PCT PSPs Banned Strict DGU at <>	PCT of PSPs banned for refusing to <i>Enable Strict DGU</i>
The following outputs are duplicated for SPs, MPSPs, MSPs, Popular Colluding MPSPs, Popular Colluding MSPs, Unpopular Colluding MPSPs, and Unpopular Colluding MSPs	
PCT PSPs Installed DGU at <>	
PCT PSPs Enabled Strict DGU at <>	

TABLE 6.7: Final List of Monitored Outputs

Initially, the exhaustive analysis of the possible factors for testing generated over 130 factors, see Appendix B. That is an overwhelming number for all types of simulation tests whether it is partial factorial, sensitivity, or sequential experimentation which we adopted in 6.5.2. For that, we neutralise many factors by fixing them in all the experiments. We anticipate that the neutralised factors won't have huge impact on the monitored outputs based on the pilot screening tests we have carried prior to starting this simulation process and based on our understanding of the nature of the implemented Algorithms. We have grouped many factors together during most of the experiments because of their strong relation and our anticipation that testing them

individually would not cause noticeable impacts on the final results. By grouping we mean that the grouped factors will be treated as a one factor. In other words, if the conducted factorial experiment requires setting the group to a high value, all the group factors would be set to high.

The tables of this section contain the final set of factors that were studied in our conducted Simulation Process. For the full list of all the factors we initially considered, refer to Appendix B. We classified the final list of factors into several groups, each group in a separate table, as following:

- **Experiment Settings:** This set of factors controls for how long an experiment runs and how many times it repeats, for better accuracy. These settings were set constant for all of the experiments. See Table 6.1.
- **Environmental Factors:** This set of factors describes how the Network bootstraps and how fast it grows. See Table 6.2.
- **Users' Controlled Factors:** This set of factors is controlled by the Users and is responsible for shaping how Users interact and react to received Abuse Cases. Most of these factors were set to constant except for the Ignorance Rate. See Table 6.3.
- **SPs' Controlled Factors:** This set of factors is controlled by the SPs and is responsible for shaping how SPs collaborate with each others and how they react to the Auditor's requests to *Install DGU* or *Enable Strict DGU*. See Table 6.4.
- **Auditor's and IDP's Controlled Factors:** This set of factors is controlled by both the IDP and the Auditor, which could be implemented as a one unit if desired. This set is responsible for shaping how the Ranks are weighted, how testing agents are created, how suspicious SPs are banned or asked to *Install DGU*, and how Ranks would be published. See Table 6.5.

- **Attackers' Controlled Factors:** This set of factors is controlled by the attacking entities and is responsible for shaping the combinations of attacks that are deployed by each different group of attackers. See Table 6.6.
- **Monitored Outputs:** This set of factors contains the main outputs described in the simulation scope 6.1. Because it is expensive to repeat a finished experiment, we tried to gather as much information out of an experiment as possible. Hence, we recorded further detailed versions of those outputs. For example, we recorded the Undetected MSPs separately from the Uncolluding Undetected MPSPs, the Colluding Undetected MPSPs, and so on. Furthermore, we have also taken different records for each measured output at four different timings, quarters, during each experiment. The reason for that is to detect any unexpected changes in behaviour that would take place as time passes. Early on the Simulation Process, we figured out that many of the monitored outputs were of less importance and, hence, we stopped analysing them thoroughly. Rather, we have kept records of them just in case a reinvestigation is needed. Table 6.7 shows the most important set of the Monitored Outputs.

6.5.2 *DOE*

There were several themes of experimental designs to consider for this Simulation Process. The sensitivity tests where only one factor would be tested at a time while the rest are fixed constant was an option that we eliminated. The reason was the fact that two-way and three-way interactions among the simulated factors would be ignored. Another option was to design a Full Factorial experiment where all the possible interactions of the factors of interest would be simulated, at different runs, to better gauge their effects. However, this approach consumes a lot of resources measuring the possible effects of interactions that we know that are hard to occur in real-life or would have little value. For that, the approach that we adopted was the Partial Factorial design where a partial set of interactions are simulated and analysed. The details of our experiment design are shown below while more technical details about the statistical concepts used in this Subsection could be found in [127].

- **Experiment Design: Partial Factorial:** a subset of the possible interactions among the simulated factors are tested at different runs.
- **Total Factors: 33.**
- **Factor Levels: 2:** Low and High values for each factor as shown in Subsection 6.5.1. This setting leads to the linearity assumption where the monitored outputs are assumed to interact in a linear manner to changes in the simulated inputs. Since linear relations are not always true, that caused minor issues that we discuss in Section 6.10.
- **Visibility: $\geq V$:** this means that the measured effects are not confounded with any two-way interactions.
- **Sample size (replicas): 30** all the runs of the partial factorial design would be repeated 30 times to reduce the bias errors. It is true that this size, which is used as a rule of thumb by some statisticians [128], may not be large enough in many cases. However, we anticipate that it would suffice our needs at this stage of simulation research because we are running a computer code that we know it would behave in quite a predictable manner and, hence, adding extra simulation runs should not generate dramatic changes in the measured outputs.

Even with a final list of 33 factors of interest, it is still not wise to simulate all those factors together in one partial factorial experiment. Unfortunately, analysing a very large experiment consumes a lot of time. Nevertheless, consuming more than 25% of the available experimental resources at the very first experiment, is normally not wise because during the experimentation process, more knowledge about the simulated factors would be gained leading to better design the following experiments to reduce their cost and improve their results quality [127]. For that, our simulation approach was to design mini-experiments where each experiment would test up to 11 factors at a time, because this is the maximum number of factors we could analyse in one experiment using our chosen statistical software, Minitab. Each experiment was dedicated to optimize a set of closely related factors. At the end of each mini-experiment, its optimized values were fed to the Simulation Model for the following

mini-experiment. Those mini-experiments were classified in 8 different Experimental Stages, see Section 6.6, where each Stage is about optimising and studying some main features of the simulated Auditor and its Counter-Attackers. Minitab software was utilized to generate the partial factorial testing plans for different factorial designs ranging from 2 factors up to 11. Those generated designs were hard-coded in a Java catalogue utilized by the simulation code to run each experiment.

6.5.3 *Statistical Analysis*

For this step of our analysis, we relied mainly on the Minitab 17 statistics software. Basically, after executing each mini-experiment, our Java model would generate a Data file containing the experiment's settings as well as the measured effects for each monitored response, see Table 6.7, for all the 30 executed simulation runs of each of the partial factorial settings, see Subsection 6.5.2. This generated Data file is then imported in the Minitab software. We then start our analysis by fitting the DoE model by specifying for the software the factors and the monitored outputs we are interested at analysing.

Once we fit the model, Minitab did automatically generate Pareto Charts of effects that show how significant the effects of each factor on each of the monitored outputs, see Minitab Support page [129] for more details. In Minitab, any effect that gets a magnitude over 2 in the Pareto Chart is considered to be of a significant effect. Nevertheless, we learned in our continuous efforts to optimize the defensive factors that the factors that barely cross the barrier of magnitude 2 would normally have little influence on our optimisation process. Hence, we adopted the following convention in describing the significance of the measured effects in Section 6.6:

- **Huge:** This refers to effects with: $Magnitude \geq 100$.
- **Significant:** This refers to effects with: $50 \leq Magnitude \leq 100$.
- **Mild:** This refers to effects with: $20 \leq Magnitude \leq 50$.

- **Ignored Effects:** These effects with *Magnitude* < 20 are not listed. Nevertheless, these factors have been considered during our analysis in the cases where there are not many more significant effects.

After determining the list of significant factors' effects, we followed by asking Minitab to generate the factorial plots showing the main effects for each factor independently as well as the effects of the two-way interactions of the simulated factors, see Minitab support pages [130] and [131] for more details. By studying these effects, we try to come up with explanations to why these factors are acting the way they are. The ongoing optimisation process helped us in confirming or rejecting some of the early explanations we came up at the beginning of the Simulation Process. This step was a crucial part of the Simulation Process because it helped us in making educated guesses on how to manually modify some of the mistakenly optimized values due to some bias or human mistakes during the process, the Manual Flavour of the Aggressive Defence Strategy we describe in Subsection 6.6.3 is an example.

A powerful tool of Minitab is the response Automatic Optimizer, see support page [132] for more details. We heavily used this tool to optimize both the Auditor and the Attackers' set of factors to reach an equilibrium point or status. That is, we normally start the process by setting the Attackers' factors to their minimum values, representing trivial attacks. Then, we ask the Optimizer to come up with the best combination of values for the Auditors' factors to fulfil the goals listed in the Simulation Scope, see Section 6.1. We then fix the optimized values of the Auditor and ask the Optimizer again to come up with the best combination of the Attackers' factors to achieve the Attackers' goals listed in the Simulation Scope. We then repeat the process with the Auditors' and Attackers' factors until we reach an equilibrium setting or a previous setting comes up again. In the latter case, it would be clear that there would be no equilibrium reached in real-life. Instead, the Auditor and the Attackers would continuously alter their settings in the quest to make the highest gain. The winner of the game would be the player that can precisely sense the settings of his opponent and counter them in time before the opponent realises the altered settings. For the purpose of our optimisation, however, we look at all the optimized scenarios

and pick the one that would give the Attackers the best outcome and choose it as the equilibrium status, since Attackers are the initiators of the game and, hence, would naturally force the Auditor to counter their attacks.

It should be noted that the Automatic Optimizer would have internal settings that enable giving differential weights to certain responses based on their importance. These weights would range from 1, default, to 10. The default weights we used in most of our experiments are:

For the Auditors' Factors

- **5:** to Minimise *Banned PSPs Rates* and *Active Agents to Users Rate*.
- **5:** to Maximise *Colluding Popular MPSPs Detection Rate* and *Users Trust*.
- **3:** to Maximise *Uncolluding MPSPs Detection Rate* and *Colluding Unpopular MPSPs*.
- **1:** to Minimise *Banned SPs Rates*, *Compromised Credentials Rates*, and *Open Cases*.
- **1:** to Maximise *Uncolluding MSPs Detection Rate*.

For the Attackers' Factors

- **5:** to Minimise *Colluding Popular MPSPs Detection Rate*.
- **3:** to Minimise *Uncolluding MPSPs Detection Rate* and *Colluding Unpopular MPSPs*.
- **1:** to Minimise *Uncolluding MSPs Detection Rate* and *Compromised Credentials Rates*.

it should be noted that after we have separated the optimisation of the Uncolluding and Colluding Attackers in Stage 1, we altered the weights so that when we optimize one group, the weights of the other groups' factors are set to 1. It should also be noted

that in Milestone 6.2, we increased the weight of the desire to Minimise the *Banned Innocent PSPs Rates* from 5 to 10 due to its large observed values at that Milestone.

Finally, once we get an equilibrium optimal setting, we would run a **Confirming Experiment**. In this Confirming Experiment, we basically fix the equilibrium settings and run the simulation with those settings for 30 runs. Then, we import the generated Data file in MS Excel so that we can take the average of those 30 runs. We used the resulted average to draw graphs showing how the measured effects would evolve at different points of times during the execution cycles. These graphs helped us at detecting potential patterns or whether a convergence in the effects is about to happen. It also helped us at detecting potential biases or non-linearity at the analysed effects of the simulated factors. That was possible by comparing the results of the final simulation with what the Automatic Optimizer had predicted.

6.6 Experimental Stages

As stated in Section 6.5.2, it is not wise to invest more than 25% of your resources in the first experiment because more understanding of the nature of the simulated system and the important factors would arise during the experimentation process, which may cause changes in the simulation plan. In our case, we started our simulation with a simple plan of 14 experiments. Nevertheless, we ended up with 33 mini-experiments that could be grouped in 8 different Stages where at the final Stage, we compare and contrast the resulted Defensive Strategies' Flavours. While the initial aim was to optimize each set of related factors at one mini-experiment and then fix them for the following mini-experiment where another set would be optimized. We thought we could further improve the accuracy of our results by re-optimising the important factors against the initial optimized system in what we called the Refinement Stages. Given the agile nature of designing and conducting this Simulation Process, there were some inconsistencies and mistakes. If this Simulation Process is to be repeated, we could design a better, more organised, approach as we explain in Subsection 6.10.3.

In this Section, we briefly present the intermediate results and early observations we have made during each of the Experimental Stages. Each of these Stages consisted of one or more mini-experiments that were analysed according to the described process in Subsection 6.5.3. These mini-experiments are further grouped into Milestones. In each Milestone, we study and/or optimize a particular basic defensive element of Section 5.1, the Attackers behaviour against a deployed Defensive Strategy's Flavour, or the effects of certain environmental factors like the nodes populations. The details of each Stage's Milestones and associated analysis are listed in Appendix C. Those intermediate results should be helpful for researchers trying to repeat our experiments or to get more insights about the nature of the simulated factors. For the final set of results, this Section could be skipped directly to the summary of the observed Defensive Strategies, their optimized Flavours, and their Counter-Attacking Strategies in Section 6.7, the Malicious Density Theory that we developed after conducting this Simulation Process in Section 6.8, and the final Evaluation of the generated different Defensive Strategies' Flavours in Section 6.9.

Note that in the following subsections as well as in Appendix C, we may describe the effect of a factor on a monitored output as positive or negative. A positive effect is not necessarily a desired goal and a negative effect is not necessarily something we are trying to avoid. These descriptions are just mathematical relations meaning an associated increase in the monitored output's value with the increase in the input factor's value for the positive relations. For the negative relations, it is an associated decrease in the monitored output's value with the increase in the input factor's value.

6.6.1 Stage 1: The no Auditor Case Followed by Initial Optimisation

This Stage started with Milestone 1.1 to test the case where there is no Auditor involved in the Trust Network to compare it to the optimized cases at the end of the Simulation Process. After that, the TLR_{avg} Approach was optimized against a set of trivial attacks and, again, against a set of real attacks in Milestone 1.2. Next, the TST Approach was added to the scene in Milestone 1.3 before ending with the introduction of the TGT Approach in Milestone 1.4. Regarding the performance of the Auditor

by the end of this Optimisation Stage, comparing the last Milestone 1.4 with the first Milestone 1.1 reveals major improvements in the *Compromised Credentials Rates* and the *Users' Trust* alongside with some slight improvements in the *Uncolluding M(P)SPs Detection Rates*. Nevertheless, the *Active Agents to Users Rate* reached 86%, which is not a good outcome.

6.6.2 Stage 2: 1st Optimisation Refinement of Attacks and Auditorial Settings

In this Stage, the aim was to refine the initial optimized version of the Auditor Defensive Strategy by re-optimising the promising factors from Stage 1. This Stage started with Milestone 2.1 where we aimed to check whether removing the *TST* Approach, or some of its selectors, in the presence of the *TGT* Approach would reduce the *Active Agents to Users Rate* without affecting the Auditor's integrity, because the *TST* Approach requires Testing Agents. Next, we decided to optimize the important factors in increasing the *Uncolluding M(P)SPs Detection Rates*, i.e. the factors belonging to either the *TLR_{avg}* or *TST* Approach in Milestone 2.2. That was followed by optimising the important factors in increasing the *Colluding MPSPs Detection Rates*, i.e. factors belonging to the *TGT* Approach in Milestone 2.3.

Interestingly, we have noticed during Milestone 2.1 a conflict of interest between the Colluding Attackers and the Uncolluding Attackers. Hence, we started from this Milestone on to optimize for each of those two groups of Attackers separately. Since the system by now started to mature with less sensitivity to variations in the already optimized factors, we decided to keep the Attacks' Settings constant and start Refinement Stages for the most important defensive factors that we observed in Stage 1. At the beginning of each Refinement Stage, we repeat the optimisation process for the Attackers to reflect how they react to our optimized Auditor. Then, the Defensive Strategy is optimized against the recently optimized Counter-Attacking Strategy. Ideally speaking, we should have started the separation of Attackers and the Refinement process from the Stage 1, see Section 6.10 for more details.

6.6.3 Stage 3: 2nd Optimisation Refinement of Attacks and Auditorial Settings

As more knowledge about the nature of the simulated model has developed so far, this Stage of 2nd Refinement should be the role model for any future Refinement Stages that might be needed, see Subsection 6.10.3. This Stage started with Milestone 3.1 trying to optimize the launched Counter-Attacking Strategy against the latest optimized Defence Strategy in the previous Stage. Then, the important factors to improve the defences against the Uncolluding Attackers were optimized in Milestone 3.2 followed by the important factors to optimize against the Colluding Attackers in Milestone 3.3.

Toward the end of this Stage, we realised that the Auditor did not get it right at this point of the Simulation Process. Perhaps that is due to non-linearity in the *PCR Threshold* and *PCR Weak Threshold* monitored responses, see Subsection 6.10.5. Based on our experience so far with the system, we anticipated that setting *PCR Threshold* to 7, *PCR Weak Threshold* to 2 and *Ignore Old G Ranks* to false is a better combination in terms of the *Uncolluding/Colluding MPSPs Detection Rates*. As a result, the next Stage is dedicated to compare two different Defensive Strategies' Flavours that we called the Automatic and Manual Flavours. The Automatic Flavour includes the settings that we obtained through the automatic optimisation so far, see Subsection 6.5.3. On the other hand, the Manual Flavour includes the settings that we thought reasonable based on our understanding of the system at this point of the Simulation Process.

6.6.4 Stage 4: Comparing the Performance of Manual and Automatic Optimisation Settings

As stated in Stage 3, see Subsection 6.6.3, we have noted that the automatic optimisation process which we have utilized so far, see Subsection 6.5.3, seemed to get confused by some bias in our experimental settings causing it to yield some counter-intuitive optimized values for some factors. By bias we are referring to the fact that

we have restricted ourselves so far to one environmental scenario, a slow bootstrapping Network environmental settings, as well as the fact that some optimized factors seems to have non-linear effects, see Section 6.10 for more details.

To mitigate for the above-mentioned limitations, we decided to manually optimize the current Defensive Strategy based on our understanding of the system so far. As it turned out later toward the end of the Simulation Process, the automatically optimized Defensive Strategy is just a Conservative Defensive Strategy Flavour that we named, for simplicity: the Automatic Flavour. On the other hand, the manually optimized Strategy turned out to be an Aggressive Defensive Strategy Flavour that we named: The Manual Flavour, see Section 6.7. Table C.1 lists the differences between the Automatic Flavour and the Manual Flavour. In Milestone 4.1 we have optimized the launched Counter-Attacking Flavours against both Defensive Flavours. In that Milestone, after we run the general experiments, varying each variables Low and High values, we used our statistical software, Minitab, to optimize the Counter-Attacking Flavours against each Defensive Flavour separately. That gave us two pairs set of optimisations: optimized Counter-Attacking Flavour against the Manual Strategy and optimized Counter-Attacking Flavour against the Automatic Flavour. Further, we used Minitab to generate two other optimisations scenarios: Trivial Attacks against the Manual Flavour and Trivial Attacks against the Automatic Flavour. We executed four Confirming Experiments for each of those four scenarios to observe the Monitored Outputs with reduced linearity bias as we now have more simulation points rather than the basic Low and High points. However, once we figured out that the Manual Flavour is the most effective defense in most of the cases, its Counter-Attacking Flavour was chosen for the following Milestone where we compare the performance of the two Defensive Flavours at different environmental settings. If this experiment is to be repeated, we should have tied each Defensive Flavour to its optimized Counter-Attacking Flavour that we found in this Milestone to get more accurate results.

Factors' Setting	Interpretation	Effects
Low <i>ini Users</i> & (P)SPs	A slowly bootstrapping Network.	Makes the <i>Banned Innocent PSPs Rate</i> so high; a side effect that is further amplified when combined with Low <i>ini Users</i> & (P)SPs and/or <i>Manual Defence</i> . However, this setting Significantly boosts the <i>Colluding MPSPs Detection Rates</i> .
Low <i>Users</i> & (P)SPs GR	A Network that is slowly gaining interest, a Network where Users have lost faith and are leaving, or simply a saturated Network.	Slightly reduces the rate of <i>Banned Innocent PSPs</i> making it slightly better for the Auditor to prefer the Manual Flavour.
High <i>ini Users</i> & (P)SPs	A popular Network among Users.	Reduces the <i>Banned Innocent PSPs Rate</i> making the Manual Flavour preferable by the Auditor.
High <i>Users</i> & (P)SPs GR	A Network that is getting popular very fast.	Reduced the <i>Active Agents to Users Rate</i> making the Manual Flavour preferable by the Auditor.
Low <i>ini M(P)SPs</i>	A Network with little malicious activities, because M(P)SPs are normally more interested in heavily populated Networks, this setting is associated with Low <i>ini Users</i> & (P)SPs and/or a highly effective Defensive Strategy making it expensive to launch malicious attacks.	Makes the <i>Banned Innocent PSPs Rate</i> so high; a side effect that is further amplified when combined with Low <i>ini Users</i> & (P)SPs and/or Manual Flavour. Nevertheless, it also significantly boosts the <i>Colluding Unpopular MPSPs Detection Rate</i> , which is usually not a major threat.
Low <i>M(P)SPs GR</i>	A Network where M(P)SPs are not eager to join the Network. That might be because the Network isn't populated enough with Users to make it attractive to take the risk of Abusing. Or, the deployed Defensive Strategy might be so powerful that the potential gained Credentials mayn't justify the taken risk.	
High <i>ini M(P)SPs</i>	A Network that is polluted with a significant M(P)SPs population. A situation that is expected when a Network is, or expected to be soon, popular among genuine Users.	Reduces the <i>Banned Innocent PSPs Rate</i> making the Manual Flavour preferable by the Auditor.
High <i>M(P)SPs GR</i>	A Network that is getting popular among M(P)SPs. That is usually expected when the Network is also getting popular among genuine Users.	

TABLE 6.8: Environmental Factors Real Life Interpretation and Effects

Populations Scenario	Optimal Defensive Strategy	Justification
Low <i>ini Users</i> & <i>(P)SPs</i> and High <i>ini Users</i> & <i>(P)SPs GR</i>	Aggressive Strategy - Manual Flavour	The Low <i>M(P)SPs Detection Rates</i> that is associated with Low <i>ini Users</i> & <i>(P)SPs</i> makes the Aggressive Strategy preferable to improve the <i>M(P)SPs Detection Rates</i> . The side effect of an increased <i>Active Agents to Users Rate</i> and the <i>Banned Innocent PSPs</i> is mildly mitigated by the High <i>ini Users</i> & <i>(P)SPs GR</i> .
High <i>ini Users</i> & <i>(P)SPs</i> and Low <i>ini Users</i> & <i>(P)SPs GR</i>	Conservative Strategy - Automatic Flavour	The side effects of High <i>Active Agents to Users Rate</i> generated by the Aggressive Strategy are magnified by the High <i>ini Users</i> & <i>(P)SPs</i> . In addition, even the <i>ini Users</i> & <i>(P)SPs GR</i> , which could reduce required Agents when they are High, are Low. Hence, the Conservative Strategy is preferred in this case.
High <i>ini Users</i> & <i>(P)SPs</i> and High <i>ini Users</i> & <i>(P)SPs GR</i> but not (Low <i>ini M(P)SPs</i> and Low <i>M(P)SPs GR</i>)	Aggressive Strategy - Manual Flavour	The High <i>ini Users</i> & <i>(P)SPs GR</i> reduces the side effects of the Aggressive Strategy making it preferable in this case. That is especially true when combined with High <i>ini M(P)SPs</i> and High <i>M(P)SPs GR</i> .
Low <i>ini M(P)SPs</i> and Low <i>M(P)SPs GR</i>	Conservative Strategy - Automatic Flavour	The Low <i>ini M(P)SPs</i> along with the Low <i>M(P)SPs GR</i> settings make the Aggressive Strategy side effects, High rates of <i>Active Agents to Users</i> and <i>Banned Innocent PSPs</i> , cost so high that the Conservative Strategy settings preferable in this case.
Low <i>ini Users</i> & <i>(P)SPs</i> and Low <i>ini Users</i> & <i>(P)SPs GR</i> but not (High <i>ini M(P)SPs</i> and High <i>M(P)SPs GR</i>)	Conservative Strategy - Automatic Flavour	The Low <i>ini Users</i> & <i>(P)SPs</i> means a very High <i>Banned Innocent PSPs Rate</i> rendering the Aggressive Strategy inefficient. That is especially true when combined with Low <i>ini M(P)SPs</i> and/or Low <i>M(P)SPs GR</i> .
High <i>ini M(P)SPs</i> and High <i>M(P)SPs GR</i>	Aggressive Strategy - Manual Flavour	The High <i>ini M(P)SPs</i> along with the High <i>M(P)SPs GR</i> settings reduce the <i>Aggressive Strategy</i> side effects, High rates of <i>Active Agents to Users</i> and <i>Banned Innocent PSPs</i> , cost making the Aggressive Strategy an optimal choice in this case.

TABLE 6.9: Optimal Defensive Strategy (Manual Vs. Automatic) at Different Populations Scenarios

So far, all the conducted Milestones were executed with the assumption of slowly bootstrapping Networks. This choice was initially made to avoid simulating very large Networks which is expensive in terms of the required resources. However, that

population setting should not always be assumed in real-life. In fact, this is the worst setting that we should try to avoid reaching because it resembles a dying Network! Hence, it is the time at this Stage to examine whether our optimisations so far would work on other environmental scenarios or not. In fact, we want to know to what extent our optimisations so far are biased to the initial assumed environmental settings.

In Milestone 4.2, the two Defensive Strategies Flavours were compared against variable initial number of nodes along with their corresponding growth rates. Table 6.8 Lists the different possible population settings in the simulation environment along with their associated real-life interpretations and their observed effects as noted after executing this Milestone. Table 6.9 also lists some of the common population settings along with the best Defensive Strategy the Auditor could deploy at such a scenario. Interestingly, we have noted that both Defensive Strategies' Detection Rates get boosted at certain population scenarios, when both *ini Users & (P)SPs* and *ini M(P)SPs* factors are set to Low in addition to setting either of *Users & (P)SPs GR* or *M(P)SPs GR* High. Setting both *Users & (P)SPs GR* and *M(P)SPs GR* High make the Defensive Strategies effects even more prevalent. This is a key observation that led us to come up with the Malicious Density Theory that is listed in Section 6.8. At the end, we found that at the slow bootstrapping scenario, which we have adopted so far in all of the previous stages, it seems that the Auditor is indifferent to either of the Defensive Strategies. Hence, we decided, from now on, to switch our assumed population to a rapid bootstrapping scenario to resemble a more optimistic real-life scenario.

6.6.5 Stage 5: Introducing DGU with Offering Voluntarily DGU Installation Randomly along with a Refinement Iteration

In the previous Stages, we have conducted exhaustive optimisation Milestones to come up with mature Defensive Strategies' Flavours that are based on Ranking Algorithms that could be implemented using the technologies available at hand today. In Stage, 6.6.4, it turned out that the Aggressive Strategy - Manual Flavour was the best choice for the Auditor to deploy in most scenarios. However, this Flavour suffers from

weak points that we wish to overcome in any real-life implementation. Particularly speaking, the High *Active Agents to Users Rate* and the High *Banned Innocent PSPs Rate* are the responses that worries us most.

In this Stage, we try to overcome the short comes of the previous Stages by introducing the Digital Governance Unit, DGU, as a tool, along with its associated Algorithms, that can be used by the Auditor to develop new Defensive Strategies' Flavours, see Section 4.4. This unit is basically supposed to be installed by (P)SPs to guarantee that they respect the sharing Sticky Policies that are attached to the Credentials they acquire from Users. Knowing who obtained a compromised Credential through sharing helps in improving the calculations of the Auditor's Ranking Approaches, see Subsection 5.6.3 for more details. It should be noted that the DGU Unit as it is described in this thesis is not ready for real-life deployment yet. The closest hardware solution to what we are describing is the TPM chip, which is still under development and faces many challenges, see Section 3.1.1 for more details. Nevertheless, our Simulation Process assumes a ready to deploy DGU and evaluates whether it is worth the efforts to develop it for the purpose of providing truly Continuous Trust Management Framework as described in this thesis.

In Milestone 5.1, we added the DGU along with its associated Algorithms, see Section 5.6.3, to the Manual Flavour that we choose in the previous Stage. Then, we optimized the Counter-Attacking Strategy against the new Defensive Strategy Flavour. Interestingly, the Automatic Optimizer for the Attackers decided to deploy a tougher Colluding Settings, preferring Strong Colluding Strategy over Weak Colluding Strategy, while relaxing their Users/Credentials Abuse Drop Rates. It turned out that the actual performance of the Attackers got worse than initially anticipated by the Automatic Optimizer. That signals a non-linearity in the measured effects of the Colluding Settings that causes a bias in our simulation, see Section 6.10 for more details.

In Milestone 5.2, we tried to optimize the important factors related to the Auditors' *TST* and *TGT* Approaches based on our experience so far with the system. The results of the optimisation were generally good. We have confirmed our belief that the

PCR Threshold factor is more effective to tackle Strong Colluding Attacks compared to the *PCR Weak Threshold* factor, which is good in Weak Colluding Attacks scenarios.

In Milestone 5.3, we re-optimized the Counter-Attacking Strategy's Flavour against the current Defensive Strategy's Flavour. It seems that the Attackers realised how bad their Colluding Settings were in Milestone 5.1 and, hence, they have undone those changes and adopted more classic settings, preferring Weak Colluding Attacks with higher Colluding Users/Credentials Abuse Drop Rates. The final results are pretty similar to Milestone 5.1, with milder effects for the Attacking Factors. That suggests the fact that the Auditors' settings are near the optimal values given the available options to the Auditor. That is especially true given that the Attackers have switched their attacking technique from Strong Colluding to Weak Colluding without significantly affecting the Auditors' performance. However, unlike our initial anticipation that the *Post DGU Install Rank* is an influential factor since it controls how much Trustworthy MPSPs look like after they choose to *Install DGU*, both Milestones 5.1 and 5.3 proved that this factor does not have any noticeable effects. Rather, we found that the advanced Colluding Settings that *Colluding MPSPs* adopt after the Auditor force them to *Install DGU* are the main factors responsible for the reduced *MPSPs Detection Rates* in this Stage.

Overall, it is true that the optimisations of this Stage have failed at achieving noticeable *Colluding MPSPs Detection Rates*, due to the Colluding Attackers deploying Advanced Colluding Attacks that we describe in Subsection 5.8.7. Nevertheless, when we compare the performance figures we achieved by the end of this Stage with those figures of the Aggressive Strategy - Manual Flavour, which is the base Strategy we used here, we find good improvements. By deploying the DGU Algorithm, the *Banned Innocent PSPs Rates* were reduced considerably from 32% to 0% as well as reducing the *Active Agents to Users Rate* from 67% in to 50%, see Milestone 5.3 and Table C.2. These improvements did not come at the cost of negatively affecting the *Users Trust* or the *Compromised Credentials Rates*. However, we got worried about the Rates of (P)SPs who *Installed DGU* and *Enabled Strict DGU*, which were above 90%. That is worrying because we do not anticipate such excitement to adopt such a technology

at a bootstrapping phase in real-life by suspecting (P)SPs who might prefer to leave the Network rather than *Installing DGU*, see Section 2.8 for more details. We call the resulted optimized Defensive Strategy Flavour of this Stage the Conservative Strategy - vDGU Flavour, where the *v* refers to the fact that the Auditor gives the option to voluntarily *Install DGU* as well as the option to voluntarily *Enable Strict DGU*. This Flavour is considered Conservative rather than Aggressive since it is more cautious in banning suspicious (M)SPs. In Stage 6.6.6, we examine the case where the Auditor do not offer the option to voluntarily *Install DGU* or *Enable Strict DGU*.

6.6.6 Stage 6: Removing the Voluntarily DGU Installation Option along with a Refinement Iteration

In this Stage, we tried to further optimize the Conservative vDGU Strategy Flavour we developed in Stage 5, see Subsection 6.6.5, but this time in an environment where (P)SPs never *Install DGU* voluntarily. As usual, we started the Stage with a Counter-Attack Refinement in Milestone 6.1, but without offering the option to voluntarily *Install DGU*. The resulted Attacks and the corresponding Auditor performance did not differ much from what we have observed toward the end of Stage 5.

Despite the unchanged behaviour even without offering the option to voluntarily *Install DGU*, we decided in Milestone 6.2 to optimize the important *TST* and *TGT* factors. That is because we were desperate to check if there are some factors or environmental settings that would do the trick and let the Auditor improve its *Colluding MPSPs Detection Rates*. Indeed, the Automatic Optimizer achieved excellent *Colluding MPSPs Detection Rates* by incorporating Weak Colluding detecting settings instead of the Strong Colluding detecting setting that were prominent in Stage 6.6.5 optimisations. However, those improvements were at the cost of very High *Banned Innocent PSPs Rates*.

In our quest to find a possible environmental scenario where utilising the powerful DGU would not cause major side effects, we conducted Milestone 6.3 with environmental settings resembling an unpopular Network among genuine Users but highly

populated and popular among M(P)SPs, see Table 6.8. In other words, we wanted to check whether the current Aggressive Strategy Flavour that utilizes the power of the DGU is the magical solution to cure Networks that are going to be dead due to being so polluted with M(P)SPs that genuine nodes would not be interested to join or utilize. Per our Malicious Density Theory, such environmental settings are probably within the Threshold Region where High *MPSPs Detection Rates* could be achieved, see Section 6.8. The resulted optimisation, however, did little improvements to the *Banned Innocent PSPs Rates* while doing worse at the *Colluding MPSPs Detection Rates*. For that, the optimisations of this Milestone got abandoned. In addition, we named the optimisations of Milestone 6.2 the Aggressive Strategy - Extreme DGU Flavour. In Milestone 6.4, we re-optimized the Counter-Attacking Flavour against the Aggressive Extreme DGU Strategy Flavour. The results show that the Colluding Attackers got extra careful by raising their Colluding Settings. That considerably reduced the *Banned Innocent PSPs Rates* and the *Compromised Credentials Rates* as well as increased the *Users' Trust*. However, the *Banned Innocent PSPs Rate* is still unacceptably high at around 18%.

Finally, we decided that the settings of the Conservative vDGU Strategy Flavour would be also utilized even if the Auditor did not offer the option to voluntarily *Install DGU*. In that case, we named the new Flavour the Conservative DGU Strategy Flavour. All the Defensive Strategies Flavours are compared against each other in Section 6.9 to check which of them would be the best option to deploy at a given environmental situation.

6.6.7 Stage 7: Comparing the performance of vDGU and DGU

This Stage consists of only Milestone 7.1. This Milestone basically compares the performance of the Conservative vDGU Strategy Flavour and the Conservative DGU Strategy Flavour, which we have developed in Stage 6, see Subsection 6.6.6, under various populations scenarios in a similar fashion to the comparison we made between the Aggressive Manual Strategy Flavour and the Conservative Automatic Strategy Flavour in Stage 3, see Subsection 6.6.3. We find out that both Flavours have similar

effects and, hence, the Automatic Optimizer did not strongly push the Auditor to deploy any one of those Strategies. That is a good result because it confirms that whether the genuine PSPs and Users love the idea of voluntarily *Installing DGU* or not, the Auditor achieves good results with a maximum penalty of *Banned Innocent PSPs Rates* reaching around 8% in the worst-case where no PSP voluntarily accepts to *Install DGU*. The results of Milestone 7.1 also confirm the hypothesis we came up with about the Threshold Density Regions in Section 6.8.

6.6.8 Stage 8: Evaluating the Different Defensive Strategies by ANOVA-Testing

In this final Stage, we identified the 7 main Defensive Strategies' Flavours that we were able to identify thorough our Simulation Process, see their summary in Section 6.7. We then simulated each Flavour, including the None Strategy, against its Counter-Attacking Strategy in different environmental scenarios. These environmental scenarios are simply the variations of 3 factors:

- **Normal Nodes Population:** This is a factor consisting of grouping the *ini Users* & *(P)SPs* and the *Users* & *(P)SPs GR* factors.
- **Malicious Nodes Population:** This is a factor consisting of grouping the *ini M(P)SPs* and the *M(P)SPs GR* factors.
- **Users' Ignorance Rate:** This factor refers to the probability of Users caring about and, hence, reporting a received Spam.

Then, we run ANOVA analysis to compare the generated effects of each Strategy's Flavours on the different monitored outputs in different environmental scenarios. The results of this final Stage should be an important guide for any wise Auditor trying to figure out the best Defensive Strategy Flavour to deploy if he roughly knows the environmental settings and the nature of the launched Attacks. While this Stages' Milestones, see Subsection C.8 for more details, lists the main notes and observations

we made during the ANOVA testing, Section 6.9 summarises these results in a tabular format.

6.7 Summary of the Optimized Defensive and Attacking Strategies

Strategy	Description	Strengths	Weaknesses
Random	Random Detection due to relying on simple Algorithms like <i>TLR – avg Approach</i> .	No Agents required; fair enough against very trivial Attacks	Worse than deploying nothing if the Attacks are complex.
Aggressive	Random Detection due to complex Attacks. Relies on Weak Colluding Detection.	High <i>Trust</i> ; Low <i>Compromised Credentials</i> ; High <i>MPSPs Detection</i>	High Agents requirements; High <i>Banned PSPs</i>
Conservative	Accurate Detection based on Strong Colluding evidence.	High <i>Trust</i> ; Low <i>Compromised Credentials</i> ; Low <i>Banned PSPs</i> ; Moderate Agents requirements	Low <i>Colluding MPSPs Detection</i>

TABLE 6.10: Classifications of the Defensive Strategies

While conducting the Experimental Stages of Section 6.6, we came to realise some Defensive patterns with associated Counter-Attacking patterns. As the Stages matured, we realised three distinctive Defensive Strategies based on those patterns as shown in Table 6.10. The Random Defensive Strategy is very simple and would be useful in a limited set of scenarios where most of the malicious nodes are expected to be novice according to the readings of the Auditor’s Health Gauges, see Subsection 5.6.4, or when the Auditor could utilize a smart Strategies’ Release Order approach to fool the Counter-Attackers to think that he is deploying an Aggressive Defensive Strategy while he could be deploying an inexpensive Random Defensive Strategy, see Subsection 7.3.1. On the other hand, both the Aggressive Defensive Strategy, that relies on Weak Colluding detection settings, and the Conservative Defensive Strategy, that relies on Strong Colluding detection settings, are powerful Strategies that we focused on optimising through our Simulation Process. Our testing, optimising, and comparing process of those main Defensive Strategies along with their Counter-Attacking Strategies, generated by the Automatic Optimizer of Subsection 6.5.3, led to many Flavours of those Defensive Strategies. It should be noted that the introduction of the hypothetical hardware Trust unit, the DGU of Sections 4.4 and 5.6.3, created more

powerful Defensive Strategies' Flavours and, hence, forced the Counter-Attackers to adopt more complex Colluding Attacks that made them almost undetectable, given the Algorithms we have proposed in this thesis. The seven main Defensive Strategies' Flavours that we have developed and evaluated are:

- **Random GPD:** This Flavour is simply about deploying only the TLR_{avg} Approach. That is, the optimized Auditor by the end of Milestone 1.2.
- **Conservative ST:** This Flavour is simply about deploying only the TLR_{avg} and the TST Approaches. That is, the optimized Auditor by the end of Milestone 1.3.
- **Aggressive Manual:** This Flavour is the result of the manually modified settings based on our understanding of the system by the time we started Milestone 4.1. See Table C.1 for more details.
- **Conservative Automatic:** This Flavour is the result of the automatically generated settings by the end of Milestone 3.3. See Table C.1 for more details.
- **Conservative vDGU:** This Flavour is the result of the automatically generated settings after deploying the DGU Algorithm and assuming some PSPs would accept to voluntarily *Install DGU*. That is, the setting we reached by the end of Milestone 5.2.
- **Conservative DGU:** This Flavour is the result of deploying the *vDGU Flavour* setting but assuming no PSPs would accept to voluntarily *Install DGU*. That is, the setting we reached by the end of Milestone 5.2.
- **Aggressive Extreme DGU:** This Flavour is the result of the automatically generated settings after deploying the DGU Algorithm and assuming no PSPs would accept to voluntarily *Install DGU*. That is, the setting we reached by the end of Milestone 6.2.

	Strong Colluding Attacks		Weak Colluding Attacks	
Description	Only Abuse a Credential after a large enough number of Colluding Attackers acquire it		Only Abuse a Credential when a large enough number of Colluding Attackers deals with the User owning it	
	Setting	Justification	Setting	Justification
Main Attack Settings	High $N_{toAbuse_C}$ value	Increase the minimum number of Colluding Attackers who have acquired the Credential	High $N_{toAbuse_U}$ value	Increase the minimum number of Colluding Attackers who have dealt with the User owning the Credential
Minor Attack Settings	Low Colluding Users/Credentials Drop Rates	To substitute the lost share of total Abuse Cases due to the strict Colluding Settings	High Colluding Users/Credentials Drop Rates	To reduce the Colluding Attackers footprints in the reported Cases to avoid getting Detected
Main Defence Setting 1	Turning on <i>PIIL GT Selector</i>	<i>PIIL GT Selector</i> is powerful at detecting Strong Colluding Attackers who appear in large Gs within the <i>PIIL</i> Record	High W_{TGT}	High W_{TGT} speeds up the banning since it biases the <i>TGR</i> Ranks toward the conclusions made by the <i>TGT</i> Ranking Algorithms
Main Defence Setting 2	Low <i>PCR Threshold</i>	Low <i>PCR Threshold</i> increases confidence in the conclusions made by the TGT_{sc} Strong Colluding Ranking Algorithm leading to faster banning decisions	Low <i>PCR Weak Threshold</i>	Low <i>PCR Weak Threshold</i> increases the confidence in the conclusions made by the TGT_{wc} Weak Colluding Ranking Algorithm leading to fast banning decisions
Minor Defence Setting 1	High <i>G Max Size</i>	High <i>G Max Size</i> helps focusing on large Gs of suspicious Colluding Attackers instead of small random Gs	Low <i>G Max Size</i>	Small <i>G Max Size</i> helps focusing on small Gs of suspicious Weak Colluding Attackers who can do their attacks without necessary appearing on the <i>MGR</i> or the MGR_{sc} Records
Minor Defence Setting 2	High W_{sc} and W_{sg}	High W_{sc} and W_{sg} make the <i>TG</i> Ranks biased toward the findings of the TGT_{sg} and TGT_{sc} Detectors which focus on Strong Colluding	High W_{wc}	High W_{wc} biases the <i>TG</i> Ranks to the findings of the TGT_{wc} Weak Colluding Ranking Algorithm
Minor Defence Setting 3			Turning off <i>Ignore Old G Ranks</i>	Turning off <i>Ignore Old G Ranks</i> helps keeping valuable MGR_{wc} logs to track the tricky Weak Colludings

TABLE 6.11: The Main Defensive Settings against the main Counter-Attacking Strategies

Regardless of the settings' differences found among the different Strategies' Flavours

we have generated, the Strategies' Flavours of each of the two main Defensive Strategies, Aggressive and Conservative, share common influential defensive settings against common influential Counter-Attacking Strategies. That is, the Aggressive Defensive Strategy try to defend against Weak Colluding Attacking Strategy while the Conservative Defensive Strategy try to defend against Strong Colluding Attacking Strategy. In Table 6.11, we briefly describe the common settings for the two main Counter-Attacking Strategies along with the main influential defensive settings used to tackle each type of those two Counter-Attacking Strategies. It should be noted that different Defensive Strategies' Flavours could try to tackle both types of Counter-Attacking Strategies, with variable bias toward one of them. Hence, the Auditor may try to generate an infinite number of Strategies' Flavours at different times depending on dynamic Attackers' behaviour that could be observed by the Auditor's Health Gauges, see Subsection 5.6.4.

6.8 The Malicious Density Theory

During our Simulation Stages, we noticed some interesting facts about the effects of the M(P)SPs ratio to the total inhabitants of the Trust Network, i.e. the Malicious Nodes Density. Based on our early observations, and confirmed by the final ANOVA comparisons that we carried for the different Defensive Strategies' Flavours at variable Density settings, we observed that:

- At the extreme case of a High Malicious Density, we observed that the Detection Rates get Low because the Ranking Algorithms cannot draw conclusions about who is colluding with who. That is caused by the enormous number of M(P)SPs involved in all the open Cases.
- At the extreme case of a Low Malicious Density, we observed that the Detection Rates also get Low because the Ranking Algorithms would confuse Innocent (P)SPs for being colluding or acting maliciously because there is a minority of M(P)SPs causing a lot of open Cases containing many Innocent PSPs appearing often together, unintentionally, giving the false sense of being colluding.

- At a Threshold Point in between the two extreme cases, we observed the Detection Rates to be higher for most of the Defensive Strategies' Flavours along with their associated Counter-Attacking Strategies.

$$D_{U,S,P} = \frac{M(P)SP_s}{(P)SP_s + M(P)SP_s + Users} \quad (6.8)$$

Equation 6.8: Basic Malicious Density Equation

$$D_P = \frac{MPSP_s}{PSP_s + MPSP_s} \quad (6.9)$$

Equation 6.9: Malicious Density Equation - PSPs

$$D_{S,P} = \frac{M(P)SP_s}{(P)SP_s + M(P)SP_s} \quad (6.10)$$

Equation 6.10: Malicious Density Equation - (P)SPs

$$D_{U,P} = \frac{MPSP_s}{PSP_s + MPSP_s + Users} \quad (6.11)$$

Equation 6.11: Malicious Density Equation - PSPs and Users

As a result of our initial observations regarding the effects of the Malicious Density, we came up with the basic Equation 6.8 to calculate the Density of the M(P)SPs among all the Trust Networks' inhabitants: (P)SPs and Users. Nevertheless, we

thought that this Equation may not give accurate results because it includes the Users' population that may not affect the integrity of Defensive Strategies' Flavours, apart from the Random Strategy - GPD Flavour, since they are not sensitive to the volume of the reported Cases by the Users. In addition, that basic Equation also includes the (M)SPs population, which may just add noise to the calculated Density values because most of the developed Strategies' Flavours are not bothered with Detecting the MSPs due to their low risk. For these reasons, we created three variations of the basic Density Equation. Equation 6.9 considers only the populations of the (M)PSPs while Equation 6.10 considers only (M)(P)SPs populations and finally Equation 6.11 that only considers (M)PSPs and Users populations.

Knowing that the Detection Rates are boosted at a given Density Region while they are slowed down in very High or very Low Density Regions, we anticipated that there is a Density Threshold Point where the Detection Rates are boosted the most. We also anticipated good Detection Rates at the Density values around the Density Threshold Point, defining what we would call from now on the Threshold Region. On the other hand, Density values Lower or Higher than the Threshold Region would lead to either of the Top/Bottom Low Detection Regions.

Density Point	Norm. Pop.	Mal. Pop.	$D_{U,S,P}$	D_P	$D_{S,P}$	$D_{U,P}$	Detection	Detection "DGU"	Notes
S1	Low	High	0.48	0.75	0.74	0.29	-	+	At edge of the Top Low Detection Region but DGU effects dragging it down near the Threshold Point.
S2	Low	Low	0.34	0.64	0.61	0.19	+	0+	Very Close to the Threshold Point but DGU effects dragging it down near the Bottom Low Detection Region.
S3	High	High	0.26	0.55	0.53	0.14	-	-	In the Bottom Low Detection Region.
S4	High	Low	0.17	0.41	0.39	0.09	-	-	In the Bottom Low Detection Region.

TABLE 6.12: The Observed Malicious Density Values during the Simulation Process

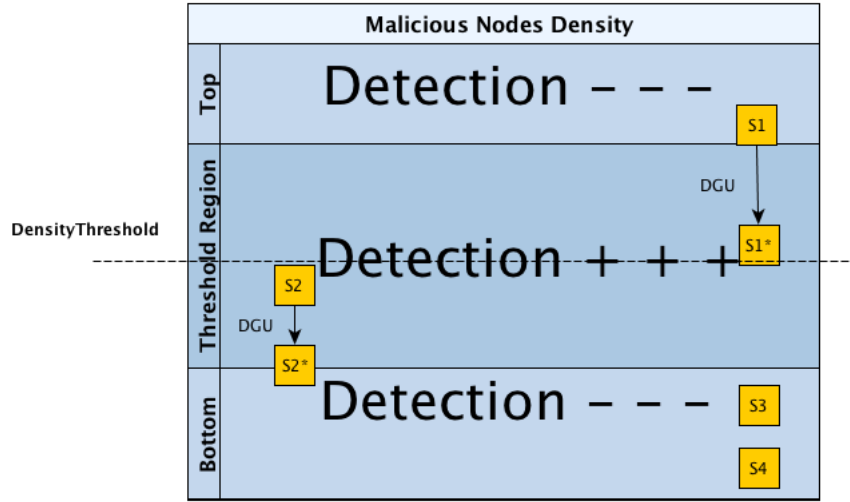


FIGURE 6.2: Visualising the Observed Malicious Density Values during the Simulation Process

To test our theory and the accuracy of each of the Density Equations, we generated Table 6.12 where we calculated the Density values for each of the Density Equations in four main Density Points. The chosen Density Points are basically all the possible combinations of the Normal and the Malicious Nodes' initial populations. The values we used in these calculations are per our standard settings we used in our Simulation Process as they appear in Table B.2. We then classified the performance of the Auditor at a certain Density Point as positive, meaning it is at the Density Threshold Region, or negative, meaning it is not in the Density Threshold Region. That classification is based on our observation of the performance achieved by the Auditor at those Points in Stage 6.6.4, for the Detection column, and in Stage 6.6.6, for the Detection with DGU column. To better visualise the generated Table 6.12 in light of the proposed theory, we created Figure 6.2.

Difference	$D_{U,S,P}$	D_P	$D_{S,P}$	$D_{U,P}$
S1 - S2	0.14	0.11	0.13	0.10
S2 - S3	0.08	0.09	0.08	0.06
S3 - S4	0.22	0.2	0.21	0.15

TABLE 6.13: The Step Value Between Consecutive Density Points Based on the Density Measure

It is important to define the boundaries of each Region and to decide on the most

suitable Density Equation among the four proposed Equations. For the first goal, we could select the S1 Density Point as the lower boundary of the Top Low Detection Region, the S3 Density Point as the higher boundary of the Bottom Low Detection Region, and the S2 Density Point as the Density Threshold Point. These boundary Points are estimates that are good enough for the purpose of our research. Further Refinement Experiments are needed to predict the exact values of those Points.

For the second goal aiming to choose the best Density Equation, we base our selection on an important quality: a large enough step size in between each Density Point. This condition is important because if the step size is small, the Regions would easily overlap making it hard for the Auditor to determine in which Region he is currently operating. Furthermore, small step sizes are prone to rounding errors and are hard for humans to comprehend. For this purpose, we generated Table 6.13 where we compared the step sizes of all the four Equations based on the Data presented in Table 6.12. Table 6.13 confirms that all of the $D_{U,S,P}$, D_P , and $D_{S,P}$ have very close values while the $D_{U,P}$ has smaller step sizes than them deeming it unsuitable measure. Since the most dominant factor among the Attackers are the MPSPs, we think D_P should be the main measure of Malicious Density. That is, the Regions' Boundary should be:

- Lower boundary of the Top Low Detection Region = 75%.
- The Density Threshold \approx 65%.
- Higher boundary of the Bottom Low Detection Region = 55%.

In our Experimental Process, see Section 6.6, we observed that some factors have some influence that could slightly shift the Regions' Boundary Points as we visualise them in Figure 6.2. We list these factors below. In Figure 6.3, we re-visualise the Malicious Density Regions based on our chosen D_P Equation. The D_P Equation is listed in that Figure as well as the different factors' expected effects on the stated Density values. An arrow down means an effect that reduces all the Density values while an arrow up means the opposite.

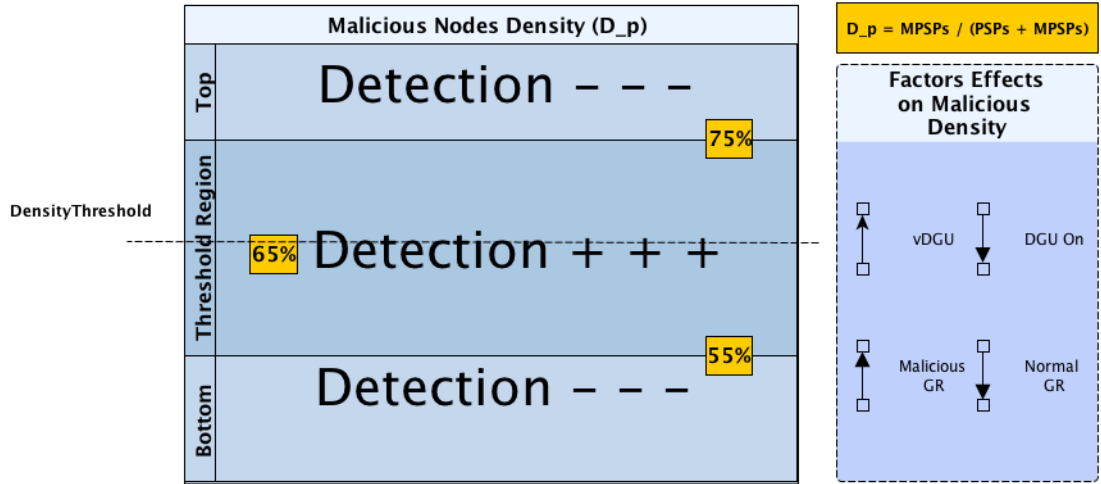


FIGURE 6.3: Approximate Density Threshold Point and Regions Boundaries

- **High *Normal Nodes GR*:** Leads to increasing the denominators of all the Density Equations. i.e. Decreasing the Malicious Density.
- **High *Malicious Nodes GR*:** Leads to increasing the numerators of all the Density Equations. i.e. Increasing the Malicious Density. Still, this increase is less significant to the opposite decrease caused by the High *Normal GR* because the High *Malicious GR* slightly increases the denominators of the Density Equations.
- **Utilising *DGU*:** Leads to increasing the Normal nodes population because it converts all the Detected Uncolluding M(P)SPs into innocent nodes. i.e. Decreasing the Malicious Density.
- **Offering *vDGU*:** Quickly removes most of the innocent (P)SPs from the reported Cases because, in our simulation settings, they quickly and voluntarily choose to *Enable Strict DGU*. Once the innocent (P)SPs get neutralised in the Density Equations, the Malicious Density would sharply increase.

A smart Auditor should take advantage of this Theory to optimize its Defensive Strategy's Flavour. That is, when it predicts that the current Density is within the Threshold Detection Region, it could deploy an Aggressive Strategy Flavour to take advantage of the current situation. When the Density gets in one of the Low Detection

Regions, the Auditor would be better off deploying a Conservative Strategy Flavour to reduce the Banned Innocent PSPs Rates. For such predictions, it is a straight forward task to calculate the denominator of the Density Equations because there is no need to distinguish between Normal and Malicious Nodes. On the other hand, it is quite challenging to accurately predict the numerator values. Some useful starting points are presented in the Auditor's Health Gauges of Section 5.6.4.

Finally, the rough Density Regions presented in Figure 6.3 may not work well for all groups of Attackers. That is, when the DGU is utilized, Colluding Attackers would resort to the Advanced Colluding settings described in Subsection 5.8.7. Once they enable that setting, it would be almost impossible to detect them given our current proposed Algorithms. In other words, they would be invisible nodes that should not be even considered in the Density calculations. Moreover, the current drawn Density Regions are potentially biased to the Weak Colluding Attacking Strategy. That is because in most of the optimisation Stages, the Automatic Optimizer choose that setting for the Attackers, see Section 6.6. After all, smart Attackers may experiment with other settings, even if such settings look less optimal, to fool the Auditor who assumes that Attackers would always deploy their default optimal settings. Altering the Attacks settings could change how effectively the Auditor Detects M(P)SPs and, hence, causes the Threshold Region boundaries to shrink, spread, and/or shift its location up or down.

6.9 Evaluating the Defensive Strategies' Flavours

	GPD	ST	Manual	Automatic	vDGU	DGU	Extreme DGU
Developed In	Milestone 1.2	Milestone 1.3	Stage 6.6.4	Stage 6.6.4	Milestone 5.2	Milestone 5.2	Milestone 6.2
Optimized in Density:	<i>S2</i>	<i>S2</i>	<i>S2</i>	<i>S2</i>	<i>S2</i> - High GRs	<i>S2</i> - High GRs	<i>S2</i> - High GRs
Auditors' Settings							
<i>W_{TGT}</i>			80%	80%	80%	80%	80%
<i>W_{sg}</i>			20%	25%	20%	20%	20%
<i>W_{sc}</i>			30%	60%	30%	30%	30%
<i>W_{wc}</i>			50%	15%	50%	50%	50%
<i>W_{TLR-avg}</i>	100%	44%	4%	16%	8%	8%	8%
<i>W_{TST}</i>		56%	16%	4%	12%	12%	12%
<i>Suspicious SP Rank</i>	32%	32%	30%	20%	30%	30%	30%
<i>Suspicious SP Banning Rank</i>	10%	10%	15%	15%	15%	15%	15%
<i>Sufficient TLRus PSP</i>	20	20	10	20	10	10	10
<i>Sufficient TLRus SP</i>	5	5	2	2	2	2	2
<i>Post DGU Install Rank</i>					50%	50%	20%
<i>Agent Report Spam</i>		false	true	true	true	true	true
<i>PSL ST Selector</i>		true	false	false	false	false	false

	GPD	ST	Manual	Automatic	vDGU	DGU	Extreme DGU
<i>Suspicious Nodes ST Selector</i>		true	false	true	false	false	true
Top TLR_{avg} ST Selector		true	true	false	false	false	false
Bottom TLR_{avg} ST Selector		true	false	false	false	false	false
<i>IIR ST Selector</i>		true	false	true	true	true	false
PCR Threshold			7	3	2	2	2
PCR Weak Threshold			2	7	2	2	2
<i>Ignore Old G Ranks</i>		false	true	false	false	false	false
<i>GT Max SP Num.</i>		2	2	2	2	2	2
<i>GT Max SP Size</i>		5	7	7	7	7	2
PIIL GT Selector			true	false	true	true	false
<i>PSL GT Selector</i>			false	true	true	true	true
<i>Suspicious Nodes GT Selector</i>			false	false	false	false	false
Counter Attacks' Settings							

	GPD	ST	Manual	Automatic	vDGU	DGU	Extreme DGU
Uncolluding Attackers' Settings							
<i>Delay Period</i>	50	50	50	50	50	50	5
<i>Bombarding Period</i>	28	28	37	37	50	50	50
<i>Drop User Rate</i>	0%	80%	80%	80%	80%	80%	80%
<i>Drop Credential Rate</i>	3%	80%	80%	80%	80%	80%	80%
Colluding Attackers' Settings							
<i>Delay Period</i>	50	50	50	50	50	50	37
<i>Bombarding Period</i>	28	28	50	50	5	5	50
<i>Drop User Rate</i>	0%	80%	75%	75%	24%	24%	20%
<i>Drop Credential Rate</i>	3%	80%	25%	25%	80%	80%	20%
<i>N_{toSpam_U}</i>	1	5	5	5	5	5	5
<i>N_{toSpam_C}</i>	0	1	1	1	1	1	5
Defensive Strategy	Random	Conservative moderated version of the <i>GPD</i> Random Flavour	Aggressive Weak Colluding focus; High <i>Banned PSPs</i>	Conservative focusing on Strong Colluding Detection	Conservative focus on Strong Colluding Detection		Aggressive focus on Weak Colluding Detection

	GPD	ST	Manual	Automatic	vDGU	DGU	Extreme DGU
Attacks Characteristics	Trivial	High-Level Weak Colluding	Similar to the Counter-Attack of the ST Flavour but relaxed <i>Drop Rates</i> to generate more Abuse				Slightly relaxed Uncolluding; Strong Colluding
Strengths	High <i>Detection Rates</i> ; Doesn't require Testing Agents	Low Agents in <i>S1</i> & <i>S2</i> ; Detects 20% <i>Uncolluding MPSPs</i> & 15% <i>MSPs</i> ; High <i>Trust</i> in <i>S1</i> & <i>S2</i> ; No <i>Banned PSPs</i>	Detects up to 40% <i>Uncolluding MPSPs</i> , 15% <i>MSPs</i> , & up to 35% <i>Colluding MPSPs</i>	Low Agents except in <i>S1</i> ; Detects up to 20% <i>MPSPs</i> & 25% <i>Colluding MPSPs</i> ; No <i>Banned PSPs</i>	Low Agents in <i>S1</i> & <i>S2</i> ; Detects up to 25% <i>Uncolluding MPSPs</i> ; No <i>Banned PSPs</i>	Detects up to 55% <i>Uncolluding MPSPs</i>	High <i>Users' Trust</i> ; Low Agents in <i>S3</i> & <i>S4</i> ; Detects over than 65% <i>Uncolluding MPSPs</i>
Weaknesses	Foiled by Simple Attacks; Low <i>Trust</i> ; High <i>Compromised Rates</i> ; Sensitive to <i>Users' Ignorance</i>	High Agents; Fails to detect Colluding <i>MPSPs</i>	High Agents	No <i>Uncolluding MPSPs Detection</i> ; very Low <i>Colluding Unpopular MPSPs Detection</i>	No <i>Uncolluding MPSPs Detection</i> ; very Low <i>Colluding MPSPs Detection</i>	No <i>Uncolluding MPSPs Detection</i> ; very Low <i>Colluding MPSPs Detection</i> ; High <i>Banned PSPs Rates</i>	No <i>Uncolluding MPSPs Detection</i> ; very Low <i>Colluding MPSPs Detection</i> ; very High <i>Banned PSPs Rates</i>
Threshold Region	<i>S1</i> & <i>S3</i>	All the same	<i>S2</i>	<i>S2</i>	<i>S1</i>	<i>S1</i>	<i>S3</i>
Densities of High Agents		<i>S1</i> & <i>S2</i>	<i>S1</i> & <i>S2</i>	<i>S2</i>	<i>S1</i> & <i>S2</i>	<i>S1</i> & <i>S2</i>	<i>S1</i>
Densities of High Trust	<i>S4</i>	<i>S3</i> & <i>S4</i>	<i>S4</i>	<i>S4</i>	<i>S4</i>	<i>S4</i>	All the same

	GPD	ST	Manual	Automatic	vDGU	DGU	Extreme DGU
Densities of High Compromised Credentials	<i>S1</i>	<i>S1 & S2</i>	<i>S1</i>	<i>S1</i>	<i>S1 & S2</i>	<i>S1 & S2</i>	<i>S1 & S2</i>
Densities of High Banned PSPs	<i>S3</i>		<i>S2</i>			<i>S1</i>	<i>S1</i>
Best in Densities		<i>S1 & S4</i>		<i>S2 & S3</i>	<i>S1 & S4</i>		
Notes	Against the predictions of the Malicious Density Theory, <i>GPD</i> works best in the Top Density Region since it is a Random Detector relying on the volume of reported Cases.	Shouldn't be deployed in a Network with High Colluding MPSPs Density. Works best in Low Detection Regions given its stable performance and moderate Agents requirements.	Powerful but its associated High <i>Banned PSPs Rates</i> makes it risky to deploy.	Fair <i>Colluding Detection Rates</i> making it a good option to deploy in the Threshold Region.	Good <i>Uncolluding MPSPs Detection</i> , bad <i>Colluding Detection</i> , but no <i>Banned PSPs</i> . Good option for <i>Low Detection Regions</i> with <i>Low Users' Ignorance Rate</i> .	Like vDGU with better <i>Uncolluding MPSPs Detection</i> in Threshold Region but High <i>Banned PSPs</i> . In <i>Low Detection Regions</i> , vDGU is preferred to get more (P)SPs to <i>Install DGU</i> .	Powerful <i>Uncolluding MPSP Detection</i> but no <i>Colluding MPSPs Detection</i> and very High <i>Banned PSPs Rates</i> making it risky to deploy in any Region.

TABLE 6.14: Evaluating the Defensive Strategies

This Section is the vital part of the whole Chapter. Here, we present a comprehensive comparison Table 6.14 where we summarise the results obtained after conducting and analysing all the Experimental Stages as shown in Section 6.6. In this Table, we start by comparing the settings of the different Defensive Strategies' Flavours that we developed during the Simulation Process, see Section 6.7. That is followed by the settings of the Counter-Attacking Flavour launched against each different Flavour. We then present our textual interpretations of those settings as well as the observed performance metrics at different Density Points as described in our Malicious Density Theory of Section 6.8. We finally list the Density Points where a wise Auditor should consider deploying each Defensive Strategy Flavour, based on our observations. Note that we have listed the factors with the most significant effects, for both the Auditor and the Counter-Attackers, in bold.

As we reach the end of our Simulation Process, Table 6.14 reveals that we have succeeded in developing Defensive Strategies' Flavours that are significantly better than the current situation, doing nothing. By better, we refer to the qualities of interest that we listed in the Simulation Scope in Section 6.1. Apart of the *GPD* Flavour, all the developed Strategies' Flavours had significantly increased the level of the *Users' Trust* and reduced the *Compromised Credentials Rates*. Generally speaking, each of the two main Defensive Strategies has its own weakness point. The Aggressive Strategy's Flavours achieve High *MPSPs Colluding Detection Rates* at the cost of High *Banned PSPs Rates* and High Agents requirements. On the other hand, the Conservative Strategy's Flavours can reduce the *Banned PSPs Rates* and the required number of Testing Agents at the cost of Low *Colluding MPSPs Detection Rates*.

Given the *Users' Trust* and *Compromised Credentials Rates* for both Strategies are similar, we recommend the Auditor to choose the Conservative Strategy to avoid populating the Trust Network with most of the artificial Testing Agents where a substantial number of genuine PSPs are mistakenly banned from the Network. Our decision is reflected in the row describing the best Density Regions to deploy each of the compared Strategies' Flavours in Table 6.14. In that row, it could be observed that there is always a recommended Flavour for an Auditor operating in any Density

Region if the Auditor is not to utilize the DGU Unit. On the other hand, there is no recommended Flavour to deploy in *S2* and *S3 Density Points* if the DGU Unit is to be introduced. In other words, the proposed Flavours utilising the DGU Unit are not recommended to be introduced in bootstrapping or saturated Networks unless it is mandatory for all the (P)SPs to install it.

6.10 Known Limitations and their Expected Effects

While conducting this Simulation Process, we suffered from limited available resources in terms of time and human power, just like any ordinary PhD thesis. Plus, the pioneering nature of this research where we are investigating new frontiers in the Continuous Trust Management Frameworks field meant the lack of similar experiments where we could get inspirations or learn some lessons. Hence, our Simulation Process has some flaws and limitations. Here we list and discuss the effects of the limitations that we are already aware of.

6.10.1 Unverified Trust Model

One of the main limitations of this research project is the lack of a verified Users' Trust Model. As we described in Section 6.4, the closest existing model to what we are doing would be the Technology Acceptance Model, TAM, along with its many extensions covering the effects of SPs perceived usefulness, easiness of use, Security, Privacy, and Trust [124–126]. Nevertheless, we decided not to incorporate this theory or any of its extensions due to the unnecessarily added layer of complexity to our code as well as the inaccuracy of the available TAM models that makes investing our resources in implementing them unjustified, see Section 6.4 for more details. Moreover, even if we have implemented the TAM model, it would lack information showing how the Users' reactions would differ after we introduce the new Defensive Strategies that we proposed in this thesis. That is, knowing that the Auditor have the utilities to detect Abusers may raise the Users' expectations and, hence, their perceived Trust would get more sensitive to unexpected Abuse incidents.

As a result, we proposed a rough Trust Model in Section 6.4 based on our assumptions about how Users behave. We based our assumptions on the current mentality where most Users simply ignore Abuse incidents and carry on without worrying too much about who Abused their personal Credentials. In fact, our model is more restrictive than that real-life scenario. That is, we assumed a 10% fall in Trust after each received Abuse and a further 1% decrease for a period of 25 simulation cycles, which could represent hours or days depending on how active a User is in the Network, or until the Auditor figures out who the Abuser was. After that period, we assume that the User would simply forget about that Abuse case and carry on.

If our proposed Trust Model is surprisingly accurate, then we do not really have a problem. Nevertheless, it is important to know how the integrity of our results would be affected if our model is not so accurate? To answer this question, we start by clarifying that we do not expect our model to be far from real-life, based on the abovementioned scenario. Maybe we just need to calibrate some factors and periods by conducting some experiments on real Users. Moreover, our Experimental Stages of Section 6.6 showed that the *Users' Trust* is always High unless we deploy a really trivial Defensive Strategy like the Random Strategy - *GPD* Flavour. That is because the Counter-Attacking Strategies against our Defensive Strategies ignore abusing most of the Users to avoid getting detected by the Auditor. In other words, we are eliminating most of the currently launched attacks by raising the game level from a defenceless Trust Network against trivial attackers to a Trust Network deploying a proactive Auditor against Counter-Attackers trying very hard to get a tiny fraction of the Abuse share they used to get before the introduction of the proactive Auditor. As a result, we are not worried that the real-life Users' Trust Model is different than what we anticipated in our experiments.

6.10.2 Limitations of the Utilized Simulation Cycle Concept

In our Simulation Process, we relied on a simulation library called PeerSim as described in Section 6.3. That library assumes the simulation model to run in a set

number of cycles representing real-life periods. That is a rough modeling of the real-life behaviour of Users because not all Users interact with the Network at the same pace. That is, while some Users interact with ten (P)SPs a day, other Users may only interact with one (P)SP on a single day. Further, even if all Users have a unified interactions frequency, we still have the problem of defining what a cycle really represents in real-life. Is a cycle an hour, a day, or maybe a week? That is particularly important to better optimize the settings of the Attacking factor *Abuse Delay Period*. In our simulation, we assumed the upper limit for that factor to be 50 cycles assuming 50 represents days. Delaying the attack for more than 50 days may lead to attacking a Users' Credential that may not exist anymore. On the other hand, if cycles represent hours, then the Attackers could easily set their *Abuse Delay Period* to 500 instead of 50.

Fortunately, we think that this limitation should not have major effects on the integrity of our research. First, even if the Users' interactions frequencies are not unified, the Strong Colluding and Weak Colluding defensive settings, see Section 6.7 that are utilized by our developed Defensive Strategies' Flavours do not rely on the wealth of the submitted Abuse cases. Rather, those Detectors depend on patterns of PSPs appearing suspiciously in different reported Cases, regardless of the frequency of their submission. The resistance of our Defensive Strategies' Flavours, apart from the *GPD* Flavour to variations in the *Users' Ignorance Rate* as shown in Subsection 6.6.8 supports this point. Second, the Experimental Stages Data, see Section 6.6, shows that the *Abuse Delay Period* has minor influence on the launched attacks. Particularly speaking, it influences the Uncolluding Attackers who are not expected to be the main threat to the Networks' integrity in most of the cases.

6.10.3 Biased Environmental Settings

When we first designed our environmental factors of interest as shown in Table 6.2 and assigned the Low and High values for the experimentation process, we tried to set the minimum values of the Attackers large enough to enable launching all different types of Colluding Attacks. Particularly speaking, we needed at least 5 MPSPs for

each type of the Colluding Attackers to avoid delaying the Attacks until the Malicious Growth Rate generates a new MPSP to enable the High-Level Colluding Attacks to take place. The number of MSPs should be larger than the MPSPs to resemble real-life situations. That led us to end-up having more M(P)SPs than (P)SPs, which basically meant that we were biased toward simulating the more pessimistic scenario: a polluted Trust Network with a majority of Attackers.

To make things more interesting, we started the Simulation Process with the minimum population settings to save on the computational resources. However, we discovered in Stage 4, see Subsection 6.6.4, that those population settings were biased toward a Low Detection Region. Hence, we tried to improve the following Stages by increasing the populations' Growth Rates. Toward the end of the Simulation Process, we discovered that the populations' Growth Rates do little to move the Simulation Process from a Low Detection Region to a Threshold Detection Region as we described in Section 6.8.

Honestly speaking, it is better to redo the optimisation process in each different Density Region to get the ultimate Defensive Strategies' Flavours for each Region based on the proposed Ranking Algorithms. Nevertheless, we think the current settings provide fair enough insights about the main characteristics of the Defensive Strategies and their main Counter-Attackers, see Section 6.7 as well as how environmental settings affect the relation between the two as described in the Malicious Density Theory of Section 6.8.

6.10.4 Awkward Experimental Stages Design

In writing this thesis, we tried our best to present the Experimental Stages and process as smoothly as possible. However, the actual execution of the Simulation Process was not smooth and straight forward at all. We were improving and editing the design on an agile basis based on our improved understanding of the system as the time passes. That is, if this Simulation Process is to be repeated, we would have better design ideas for it.

First, the technique we utilized to conduct Stage 1, see Subsection 6.6.1 of our Simulation Process was simply to optimize the relevant Auditor's and Attackers' settings at each Milestone. That meant spending a lot of time on analysing each Milestone while, at the same time, reducing the accuracy of each Milestones' results due to the increased number of evaluated factors. From Stage 2 on, see Subsection 6.6.2, we started each Stage by optimising the Attackers' settings at a dedicated first Milestone and then optimising the Auditors' settings against the initial optimized Attack in separate Milestones. The settings may not be accurate enough that way, hence, we repeated that process for several Stages in what we called the Refinement Stages, to get stable realistic results. Due to time limitations, we could not run Refinement Stages after we introduced the hypothetical DGU Unit. Still, our findings should be good enough at this phase to understand the possible effects of this unavailable technology yet.

Another issue with the Experimental Stages design is the fact that we initially assumed the Attackers acting as a one group with similar interests when we tried to optimize their attacks in the first Stage. Nevertheless, we came to realise that there is potential conflict of interest in between the Colluding and Uncolluding Attackers as we showed in Stage 2, see Subsection 6.6.2. From that Stage on, we started to optimize for each group of Attackers separately. Although the first stage had the two groups tied together, we do not expect that to have a major threat to the integrity of our findings due to the utilisation of the Refinement process that aims to eliminate inaccuracies in the prior Stages.

6.10.5 Non-Linearity in the Measured Effects

We showed in Section 6.5 that our Simulation Process is based on conducting sequential Stages of mini-experiments. Those mini-experiments were designed based on the partial factorial principles with two simulation points: High and Low. That is, we measured the effects of each factor when its value is Low and when its value is High. Then, our statistical software, Minitab, draws a line between the two measured values of each response to understand the effects of varying the simulated factor.

While this simple approach gives us valuable insights about the potential effects of each simulated factor, it has the risk of drawing linear conclusions about non-linear relations. That is, for example, many factors could have no significant effects on the measured responses they generate until their value reaches a certain threshold near the High value. To capture such behaviour, it is recommended to introduce centre points in the partial factorial designs in addition to the Low and High points to better draw the effects graph, see [127] for more details.

However, introducing centre points in our design wasn't feasible. That meant increasing the required time to run each experiment by around 50% for each additional centre point. In addition, that also meant radical changes to our hard-coded Java catalogue of partial factorial designs used to run our mini-experiments, see Subsection 6.5.2.

Given the nature of this Simulation Process, the cost of the extra accuracy generated by the introduction of centre points is unjustifiable at the moment. Instead, we resorted to the Confirming Experiment as a screening tool to confirm whether the automatically optimized values generate the expected effects or not, see Subsection 6.5.3. Significant deviations from the predictions of the Automatic Optimizer probably signal non-linearity in the measured effects in the first experiment. That is because the Automatic Optimizer makes its predictions based on the linearity assumption. While most factors did not show significant deviations from the linearity assumption, few important factors did. Particularly speaking, the *PCR Threshold*, *PCR Weak Threshold*, $N_{toAbuse_U}$, $N_{toAbuse_C}$, and W_{TGT} factors showed non-linearity behaviours as can be seen in Subsections 6.6.3, 6.6.5, and 6.6.6. To mitigate the adverse effects of the non-linearity showed by those factors, we tried to repeat the Confirming Experiment with manually adjusted values for those factors to get more realistic optimized values. Nevertheless, future optimisation efforts looking to generate better accurate optimisations should consider introducing centre points for those important factors.

A final note is about the potential non-linearity observed in Milestone 5.2 in the Colluding Settings that caused the Attackers to focus on the Strong Colluding Attacking Strategy instead of the potentially more rewarding Weak Attacking Strategy. Instead of manually adjusting the optimized values for the Attackers, we decided to keep it as

it is because we wanted to observe how the Auditor adjusts its Defensive Strategies against a Strong Colluding Attacking Strategy. At the end of the Stage, the Attackers readjusted their Attacks automatically to focus again on the Weak Colluding Attacking Strategy. That is, by ignoring to promptly fix the non-linearity possible flaw, we were able to observe an interesting possible real-life scenario. After finishing our analysis, the Attackers' Automatic Optimizer realised its mistake and readjusted its values without our intervention. The final results of our observations about the Strong Colluding Strategy and Weak Colluding Strategy could be seen in Section 6.7.

6.10.6 Programming Errors

This Simulation Process required enormous amount of agile, incremental improvements, coding for both the Simulation Model, see Section 6.3, and the Simulation Engine that is based on the Peersim library, see Section 6.3 for more details. By enormous, we are talking about over 24,000 lines of Java code without counting empty lines. We do not expect our code to be perfect because doing this large coding project without formal verification is prone to mistakes.

One error we have discovered toward the end of our Simulation Process is the fact that the calculated *Compromised Credentials Rates* do only count the Compromised Credentials that the Users have actually reported to the Auditor. That is against our intention to count all the Compromised Credentials including those that the Users decided not to report. Nevertheless, this error is not a big deal since the *Users' Ignorance Rate* is known to be either 20% or 80% depending on the simulation settings. Hence the current *Compromised Credentials Rates* could be increased by 20% or 80% of their currently recorded values for a good approximation of the real Rates.

Another error we have discovered earlier in our process is the fact that a set of measures we have coded to count the percentages of the MPSPs each Ranking Algorithm has banned, forced to *Install DGU*, or forced to *Enable DGU*, is not giving accurate results. Although we have figured out this error early in our Simulation Process, we decided to ignore it and continue conducting our experiments without utilising those

measures on our analysis. They would have been very helpful and insightful if they were working properly, but we could not afford the time and resources to fix them.

6.10.7 Human Mistakes in Updating the Experiments Settings

Like the programming errors, human mistakes were present while feeding each experiment's Configuration file with the optimized settings of the Automatic Optimizer, see Section 6.5.3. Usually, finding these mistakes meant repeating all the experiments that were based on the wrong inputs to insure the results' integrity. This hard task was not always possible specially when we discover an early Stages' error toward finishing the Simulation Process; it would have meant basically repeating the whole process!

Among those mistakes is the fact that the Milestones starting from Milestone 1.3 until Milestone 2.1 were conducted with an *ini Uncolluding MPSP* = 1 instead of 3 like the rest of the following Milestones until Milestone 4.2. Repeating those Milestones would have been very expensive since we found about it toward the end of the Simulation Process. We anticipate that this mistake has negatively affected the automatic optimisation process for Defending against the Uncolluding Attackers in Stage 1, see Subsection 6.6.1. However, given the following refinements Stages and the Manual edits we have introduced in Stage 4, see Subsection 6.6.4, the negative impacts of this mistake should be substituted.

Another mistake occurred in Milestone 5.3 and lasted until the end of the Simulation Process. It is the fact that we input the values of $W_{TLR_{avg}}$ and W_{TST} as 8% and 12% instead of 4% and 16%. That basically meant that the Final *TG* Ranks were slightly more biased toward the decisions of the *TLR_{avg}* Approach at the cost of the decisions of the *TST* Approach. Given the small differences in the values and given the Low influence of the two factors on the measured effects during the whole Simulation Process, we thought fixing this mistake does not worth repeating the last 3 Stages.

Chapter 7. Discussions and Future Prototype Enhancements

In this Chapter, we discuss the whole picture of the conducted research in this thesis. We start the discussion by arguing that our proposed Continuous Trust Management Framework does achieve its original goal, satisfying all its Stakeholders, by providing the vitally missing feature in the current Trust Frameworks Continuous Data Control where the exchanged Data would be handled in a Trustworthy manner before and after the Data release to the other parties. We support our argument by fitting our developed and simulated prototype at the core of our proposed Continuous Trust Management Framework Design of Chapter 4. That is, we answer the questions of whether our proposed Design would satisfy the needs of all its Trust Stakeholders and whether it is practical for development at the near future given the current state of the art technologies combined with our proposed Auditor Model.

The Continuous Trust Management Framework may not be developed completely as it appears in Chapter 4 sometime soon. Nevertheless, the proposed ideas, algorithms, and future milestones should be vital to make this Continuous Framework a reality. The performance of our prototype has raised the level of the game between the Trust Framework trying to protect its Users' Data against the malicious attackers trying to compromise those Credentials. That means making the Data breaches more expensive for the attackers and, hence, reducing their volume leading to increased Trust among the Network's Users. While our proposed Auditor is good at tackling Basic Active Attacks against the Users' Credentials, see Subsection 4.2.2, it is still challenging to fully protect against some types of Data attacks like the Targeted 2.0 Attacks or the Passive Attacks, see Subsection 2.4. Also, while the proposal relies on Security features built in the state of the art technologies, it is essential to evaluate whether the Framework's components would work in harmony or whether new loopholes would arise by the new combination.

Also in this Chapter, we evaluate the proposed Continuous Trust Auditor prototype in Chapter 5 to check its resilience against the posed risks listed in the Threat Model of Section 5.8. This evaluation is based on the observed performance of the Simulation Process described in Chapter 6. Finally, we list some possible future enhancements to the current prototype to improve its performance based on our discussions.

7.1 Discussing the Proposed Continuous Trust Framework, Should it be Trusted?

In this Section, we try to answer the most important question, should our proposed Continuous Trust Management Framework be Trusted? That is, is it able to offer the essential Trust Requirements we list in Section 2.5 for its Stakeholders? Moreover, is our proposal practical to deploy given the available technologies and the efforts needed to bootstrap it? We believe that the answer to the previous question is generally, yes! It is true that the proposed prototype is far from ready for deployment as it needs the addition of some essential building blocks as well as further pilot studies. Nevertheless, our research addressed many important Trust issues as we discuss in here.

7.1.1 Satisfying the Trust Stakeholders Needs

A main design principle that influence our proposed Continuous Framework Design is to consider the needs of all the main Stakeholders as we describe in Subsection 2.2. As a result, our Continuous Trust Framework Design, see Section 4.1, incorporates some of the best building blocks that are available in the existing Trust Frameworks. Particularly speaking, our proposed Continuous Framework Design contains Authentication, Authorisation, and Secure Communication layers to insure the Security of the Continuous Framework. The Trust Management layer would enable negotiating tailor made contracts by means of Sticky Policies, see Subsection 4.4.1. Further, the Auditor unit would enforce such contracts, see Subsection 7.1.2, and maintain access logs that could be presented, upon request, to Legal Authorities in case they need to proceed their own investigations. In other words, our proposed Continuous Trust

Framework Design should satisfy all its Stakeholders since it considers all the Trust Requirements we describe in Section 2.5. It is true that the current prototype does not implement all the components present in the proposed Continuous Trust Framework Design but, as we discuss in Subsection 7.1.3, this task should not be very challenging for a group of expert programmers.

It is true that our prototype currently focus on the interests of one Stakeholders, the User. We think that is important because most of the legacy systems already focus on protecting the rights of the Service Providers by paying great attention to the Security issues. Improving the Network's Security is not a bad thing but it should be accompanied by improving the Users' Privacy to avoid letting down an important Trust Stakeholder, the User. As we show in Section 4.2, our prototype is built on top of the OpenID Connect protocol that takes care of implementing the Secure Communication, Authentication, and Authorisation layers of our proposed Continuous Framework Design. That is, we are already taking care of the Service Providers' most important concern: the Network's Security. Nevertheless, we need to conduct integration tests to ensure that the combination of protocols and algorithms would not generate new loopholes in the Trust Framework.

When it comes to Legal Authorities interests, the Auditor Unit would be a welcomed addition since it could reduce the volume of Abuse Cases they would have to investigate manually. In case they need to carry out a manual auditing, the kept Records by our Auditor would provide a great help for their task. Nevertheless, pilot real-life experiments should be carried to verify the extent of help our prototype could provide for the Legal Authorities.

7.1.2 Enforcing Continuous Data Control

In Section 3.5, we illustrated that the main weakness of the current Trust Management Frameworks is the lack of support for a Continuous Data Control Attribute, especially after their release. Hence, the main focus of our proposed Continuous Trust Management Framework is to provide that vital feature. That is, to protect

the Users' Credentials that they share with other parties through Trust transactions before they release them as well as after they release them. In Chapter 5, we introduce a set of Ranking Algorithms, Utility Units, and Deployment Rules that should aid the Auditor Unit of our proposed Continuous Trust Framework Design in detecting malicious entities abusing the acquired Data after their release. In our Simulation Process, see Chapter 6, we developed several Defensive Strategies' Flavours that deploys the different configurations of the proposed Algorithms and Units of Chapter 5 based on the Network's environmental situation. those Defensive Strategies Flavours were able to achieve significant improvements in terms of maximising the *Users' Trust* and minimising the *Compromised Credentials Rates* when comparing these metrics to the widely deployed Strategy today: doing nothing! Although some of our developed Defensive Strategies Flavours assumed the presence of the DGU unit, even the Flavours that did not have that assumption achieved significantly good performance, see Section 6.9.

It is true that the wealth of the generated Abuse Cases for some Credentials in some types of attacks, like credit cards stealing attacks, could be much less than the wealth of Abuse Cases generated by other forms of attacks, like email spamming because the email address is a frequently used Credential and the spam is an easy way to signal the Abuse Case. Nevertheless, most of our Defensive Strategies Flavours showed immunity toward High *Users' Ignorance Rate* when it comes to reporting Abuse Cases. In other words, even when we reduce the flow of Abuse reports, in analogy to the situation with other Credentials types where it is hard to detect Abuse, our Defensive Strategies Flavours would still be able to provide good protection.

When it comes to enforcing after release protection on Data sets rather than the Credentials, many challenges that are not addressed by our prototype would arise. First, it would be essential to identify the proper Data owner before trying to protect his rights. That could be an easy task when it comes to Data files like pictures or videos taken by the owner's camera. Such files could be easily watermarked with the date of creation to prove ownership of it in a similar fashion to the UCON projects, see Subsection 3.1.2. Nevertheless, watermarking blog's posts or products' reviews

would not be a practical option. Second, looking for Data Abuse Cases could face performance issues. Unlike the User's Credentials, which is normally a limited set, the Data generated by the User is enormous. If every User would deploy bots traversing the Internet looking for fractions of his Data that might have been Abused, the Trust Network would probably get congested by such bots' traffic. The Data Classifier unit in our proposed Continuous Trust Framework Design would reduce the severity of this problem as it would enable the Users to classify the important Data that they require to impose protection on them. The UMA project is aimed at managing Users' Data sets and could help at the task of providing intuitive Data classifications, see Subsection 3.1.6.

The hypothetical DGU unit that we present in Section 4.4 could further help at the issue of protecting the Data after release. If all the Trust Network residents agree to *Install DGU*, then the DGU would be able to keep Trusted Records of all the Data generated by any User, on the User's machine, even if that Data is a social media comment or a sent email message. Further, the DGU proxy would not let any Network entity to download any piece of Data unless this action is allowed by its attached Sticky Policy. This feature is not only important for the Users, but also for the Service Providers who are eager to protect the digital rights of their Data, see Subsection 3.1.1.

It is true that our prototype would work best when protecting against Mass Active Attacks. That is, our Algorithms are not designed and should not be assumed capable of protecting Users' Credentials against Passive or Targeted Attacks. That is because our Ranking Algorithms depends on analysing access logs generated after receiving an Abuse report. The more received reports the Ranking Algorithms get, the more evidence it would have to deduce who is the Abuser. Passive Attacks do not generate Abuse reports while Targeted Attacks do not generate enough reports, see Subsection 2.4.

Nevertheless, we think if the hypothetical DGU unit becomes a reality and adopted by all the Trust Network's residents, it would help in tackling these issues. That is, the DGU proxy unit would monitor all incoming and outgoing traffic in any entity.

In that case, if a malware is installed in the machine of a Targeted Attack victim, the malware would not be able to transfer any classified Data outside the machine as it would be blocked by the DGU proxy, see Section 4.4. Further, even if an MSP acquired some Data through genuine interaction, he would not be able to blackmail her owner. That is, the identity of the MSP would be verified by the DGU that he already installs. That means the victim could report the blackmail attack to Legal Authorities who would figure out who is the MSP and take proper action against him. Further, the User would be able to use the revocation feature to revoke the given grant for the Abuser to access the Data. The DGU would be able to immediately erase such Data instantly from the Abuser's machine. Even if the MSP tries to share this Data with another MSP that does not *Install DGU*, the DGU may not allow this transfer if the Data Sticky Policies' prevent this sharing.

7.1.3 Enabling Real-Life Deployment Given the State of the Art Technologies

We believe that a dedicated team of expert developers should not have great problems to make our proposed Continuous Trust Framework Design, see Section 4.1, a reality in a relatively short period of time. That is, our presented prototype in Section 4.2 already deploys the main building blocks for the Communication Channel, Authentication, and Authorisation layers. Our Auditor's Algorithms of Chapter 5 are already coded in our Simulation Model of Section 6.3. That is, it is possible to reuse the code to generate a real Auditor instead of a simulation unit.

Generalising the provided protection from the specific spam use-case that we implement in our prototype to any Users' Credentials could be tedious, but straight forward since the mechanism is the same, see Subsection 7.1.2. Building proper negotiation unit and some of the mentioned enhancements of Section 7.3 could get tedious as well but, again, straight forward given the theories are already present. Implementing the DGU, on the other hand, would be a real challenge that may not be ready in a short period of time. Nevertheless, our developed Defensive Strategies achieved significantly good performance even without utilising the DGU, see Subsection 6.7. Hence, it is

still possible to introduce the proposed Continuous Framework without the DGU unit. DGU units could be bootstrapped at a later phase once the Trust Network's residents start to build Trust on this Framework.

When developing a real-life implementation of our proposed Continuous Framework, developers should try to keep an eye on some worrying performance metrics. First, the Aggressive Strategy's Flavours, see Section 6.7, generate an unacceptably High *Active Agents to Users Rate* reaching around 80%. That is, a Network full of artificial undercover Users, or simply Testing Agents. Such a Network would cause real Users and SPs to leave while MSPs could take advantage of the situation by making money out of the Testing Agents pocket, see Subsection 5.8.3. The current solution is to refrain from deploying this Strategy and relying instead on the Conservative Strategy's Flavours like the Automatic Flavour that have moderate, but still worrying, *Active Agents to Users Rates*. The proposed enhancement of Subsection 7.3.9 could help in improving this situation, but that would require further studies to confirm.

Finally, one weak point of our current prototype is the fact that it does not address the threat of malicious Auditor or IDP. We assumed these entities to be fully Trusted while, in reality, that is not necessarily true. As we describe in Section 4.4, this issue would be tackled if these entities deploy the DGU hypothetical unit. Given this unit's expected absence in the near future, the Continuous Framework developers should resort to temporary mitigations to avoid creating a single point of failure for the proposed Framework. A possible mitigation would be to have multiple Auditors and IDPs to operate independently in the Trust Network. The entities of the Network would then rank those Auditors and IDPs based on their experience. Bad Auditors or IDPs should get lower Ranks leading to less reliance on them. Maybe some Network's residents could create their own Testing Agents and Testing SPs to interact with the suspicious IDPs and Auditors to prove whether their Ranks are correct or not. Such a complex Ranking Algorithm would re-introduce the threats we described in Section 5.8 but, this time, for malicious and colluding Auditors and IDPs instead of SPs. Given the extra powers the Auditors and IDPs have compared to the SPs, the threat model should be expanded accordingly. Otherwise, the Network residents should put

their sole Trust on an official Auditor that is run by a government or an institute with a good reputation. At the end, most of the current Trust Frameworks would operate assuming a Trusted operator.

7.2 Discussing the Extents of the Posed Risks in the Threat Model

In this Section, we evaluate the proposed Continuous Trust Auditor Model of Chapter 5 to check its resilience against the posed risks listed in the Threat Model of Section 5.8. This evaluation is based on the observed performance of the Simulation Process described in Chapter 6.

7.2.1 *Effects of Users Not Reporting all Abuse Cases*

In Section 6.9, we show that the Defensive Strategies' Flavours that we have developed are all resilient to fluctuations in the *Users' Ignorance Rate* to report received Abuse Cases. The only exception is the Random Strategy - *GPD* Flavour, which is not a good Strategy to deploy by a wise Auditor in most of the cases, see Subsection 7.3.1 for the few cases when it would be wise to deploy such a Defensive Strategy. Hence, this threat does not pose a major risk to the integrity of our proposed Auditor in the case of protecting Users against Active Attacks Abusing, the use-case that we have simulated.

Nevertheless, utilising our Strategies to enforce the Sticky Policies attached to Credentials or Data Sets in other use-cases may reveal more sensitivity toward *Users' Ignorance Rate*. That is especially true in case of Passive Attacks or Targeted 2.0 Attacks, see Section 2.4, where the attackers would barely leave any footprint revealing the existence of their malicious activity, let alone their identity. However, such cases would be viewed by Users as more serious breaches, in comparison to the more common spamming attacks for example. Hence, we anticipate that the *Users' Ignorance Rate* in such serious cases would be Low. In Subsection 7.3.7, we describe a possible future enhancement that would make reporting breach Cases more User friendly and, hence, reduces the *Users' Ignorance Rate*.

7.2.2 Effects of Malicious Users

This threat was not considered in the Simulation Process, which we describe in Chapter 6, due to time limitations. It is almost impossible to detect malicious Users launching Passive Attacks, i.e. interacting genuinely with other entities while analysing the published *TG* Ranks by the Auditor, see Section 5.5, against their own reported Cases to Reverse-Engineer the Auditor's Algorithms and update the ME that they are associated with, see Subsection 5.8.5. Nevertheless, utilising a User Index utility, similar to the proposed enhancement in Subsection 7.3.7, could eliminate the number of possible malicious Users in the Network and, hence, their power to tamper with the Auditor's integrity.

7.2.3 Effects of Sharing with Malicious SPs

Although our Simulation Process, which we describe in Chapter 6, includes sharing partnerships among (M)(P)SPs, we were not interested in studying the effects of those partnerships due to time limitations. Nevertheless, if the rate of Data exchange among the partners and/or the period of the partnership are Low, then there would not be enough footprint to detect either partnership. On the positive side, the amount of generated Abuse Cases due to such partnerships would be minimal due to the Low exchange rate among the two partners.

On the other hand, if the exchange rate among partners and/or the partnership period is High, then they would appear in the *MGR* Records, see Section 5.4, as colluding partners. As a result, they could be both banned which would be unfair for the innocent partner. Nevertheless, the Defensive Strategies' Flavours that utilize DGU would give any innocent partner the chance to prove their integrity by agreeing to *install DGU*.

When it comes to Credentials or Data Sets other than our simulated email address use-case, this threat should not be a major issue. In real-life scenarios, genuine PSPs would not share Users' sensitive Data with another (M)PSP since the Users would

assign such Data a strict Sticky Policy preventing its sharing, see Section 6.4. Even if a User did not assign such a Sticky Policy to her sensitive Credentials, it is not imaginable that an innocent PSP would share his Users' Credit Cards numbers with another (M)PSP. In case of less sensitive Data, they would probably be treated similar to the simulated email address and, hence, this threat would have the same Low impact on those other Credentials.

7.2.4 Effects of Reverse Engineering the Detection Algorithms by a Malicious Entity

While conducting our Experimental Stages, see Section 6.6, we have noticed that in most of the simulated scenarios, it is rare to find an equilibrium status where both the Auditor and the Counter-Attackers would be happy with optimal settings. Rather, each entity is continuously altering its settings to maximize its gains based on the latest settings made by its opponent. Actually, when the Automatic Optimizer described in Subsection 6.5.3 optimizes the settings for one entity, it has full knowledge about the exact settings of that entity's opponent, which enables the best possible optimisation. In contrast, such full knowledge cannot be assumed in real-life and, hence, it is vital for the Auditor to rely on a set of robust Network Trust Health Gauges as described in Subsection 5.6.4. For the Counter-Attackers, it is equally vital to rely on robust Reverse Engineering tools to better respond to updates made to the Auditor's Defensive Strategy Flavour.

The entity with the most accurate tools to analyse its opponents' settings is the entity that would make the most gains. That is, if the Auditor, for example, noticed that most of the current Attacks are launching Strong Colluding Attacking Strategy, see Section 6.7, it could deploy a Defensive Strategy Flavour that focuses on Strong Colluding detection. If the Attackers could quickly enough figure out the nature of the newly deployed Strategy's Flavour, they could switch to the Weak Colluding Strategy. Only when the Auditor realises the changed nature of the attacks, it would deploy a Weak Colluding oriented Defensive Strategy Flavour, see the proposed enhancements of Subsection 7.3.1 for more details about how to dynamically update the Auditor's

deployed Strategy. However, the Attackers would have a great advantage as long as the Auditor does not notice its altered Counter-Attacking Strategy.

When it comes to malicious Users cooperating with the Attackers by interacting in certain patterns to understand how the internal ranking works, this would be an unpractical threat unless when deployed in bootstrapping Networks. While malicious Users are almost impossible to detect, the User Index, proposed in Subsection 7.3.7, should minimise their influence on the decisions of the Auditor and, as a result, minimise the amount of knowledge they can infer about its deployed Algorithms. Further, applying such a technique to a mature Network is not expected to generate a good model since the majority of residents, which are not affiliated with the malicious entity, would have greater effects on the decisions of the Auditor meaning that the reconstructed model would be affected by Network noise. However, if such a technique is utilized in a fresh new Network where the malicious Users and SPs belonging to the malicious entity could form the majority of the Network's residents, then the reconstructed model would be very accurate.

There is also the threat of large number of temporary MSPs indulging in malicious activities to get caught and, hence, enable its creator to infer the detailed internal settings behind their detection. We think this is not a major threat to worry about because no wise ME would sacrifice an MPSP for the sake of inferring the internal settings of the Auditor. Rather, if this technique is to be used, the ME would probably sacrifice temporary MSPs. Our Simulation Process reveals that MSPs are not a major threat to the Trust Network and, hence, most Defensive Strategies' Flavours do not even bother at detecting them. That would mean even if the ME is able to get a good understanding of the internal defensive settings behind detecting its temporary MSPs, that would be of little value since it would be different than the defensive settings used to detect the more important MPSPs.

When it comes to reverse engineering the deployed Defensive Strategy's Flavour by monitoring how the published *TG* Ranks would react to alterations in the Attacking settings, the case is trickier. If the response is harsh, then that would signal for the Attackers that they should change their deployed attacks. Hence, it would be

wise to delay publishing the *TG* Ranks to obscure the Attackers' analysis and slow down their optimisation process. Nevertheless, the more the publishing of the *TG* Ranks is delayed, the more Users would deal with the Attackers assuming they have High *TG* Ranks. This trade-off should be carefully balanced through further future experimental studies as proposed in the Enhancement of Subsection 7.3.5.

7.2.5 Effects of Misleading the Auditor with Simple Data Abusing Algorithms

Our Simulation Process revealed that the Simple Data Abusing Algorithms, described in Subsection 5.8.6, would be the main weapons available for the Uncolluding MPSPs. When the Auditor deploys a trivial Defensive Strategy Flavour like the Random Strategy - *GPD* Flavour, the Uncolluding MPSPs would have to raise the Abuse Delay and Bombarding Periods for moderate values to fool the Auditor. However, when the Auditor deploys a more advanced Defensive Strategies' Flavours, especially Aggressive ones, then the *Abuse Bombarding Period* would be of little help to protect the Uncolluding MPSPs. Setting the *Abuse Delay Period* to its max value would provide some help. Nevertheless, the most vital weapons for the Uncolluding Attackers would be the *Users/Credential Abuse Drop Rates* that should be set to their max values to minimise the MPSPs Detection Rates. Still, in most of the Density Points, see more about the Density Points in Section 6.8, the advanced Defensive Strategies' Flavours would be able to achieve at least 20% *Uncolluding MPSPs Detection Rate* even if all the Simple Data Abusing Algorithms are set to their maximum values, see Section 6.6.8. In fact, the Aggressive Strategy - Extreme DGU Flavour would always achieve around 70% *Uncolluding MPSPs Detection Rates* regardless of the Density Point it is operating at.

These Simple Data Abusing Algorithms would also be helpful for the Colluding MPSPs but not to the same extent. That is, both the Strong Colluding and Weak Colluding Attacking Strategies would rely mainly on their Colluding Settings. In case of the Strong Colluding Strategy, the Attackers would slightly relax their *Users/Credentials Abuse Drop Rates* to substitute the number of Abuse Cases they would generate in

exchange for their restrictive and limited Colluding Settings, see Section 6.7. On the other hand, in case of the Weak Colluding Strategy, the Attackers would be better off increasing their *Users/Credentials Drop Rates* to decrease their footprint given their relaxed Colluding Settings. The *Abuse Delay Period* would mostly be irrelevant since both Colluding Strategies would mean naturally waiting for long times until the victims has dealt with the minimum number of Colluding Attackers per their Colluding Strategy for the attack to take place. This long time is usually close to the maximum value of the *Abuse Delay Period*.

High *Users/Credentials Abuse Drop Rates* would mean that the Attackers would refrain from attacking most of the Users in fear of looking suspicious and, hence, banned from the Trust Network. That is a desired outcome since it would mean a more Trustworthy Network. Our Defensive Strategies' Flavours, apart from the Random Strategy - *GPD* Flavour show some immunity against High *Abuse Delay Periods*, which is also good. However, we are not sure what would happen if the Attackers decide to raise the maximum value of the *Abuse Delay Period* from 50, like in our experimental settings, to a larger value like 500. Maybe that could success in fooling our Defensive Strategies' Flavours to some degree. Nevertheless, the larger the *Abuse Delay Period* is, the less value the Attacker would get out of the Abuse. That is, if the period resembles days in real-life, Abusing an email address after 500 days of acquiring it may reveals that the User has already dismissed that email address and, hence, the Attacker would get no gain out of this attack. Similar scenarios would happen when attacking a Credit Card after its expiry date or when blackmailing a User who have already passed away! In order to better understand the real effects of this type of attacks, we should build a more accurate Simulation Model with a more real-life interpretation of the execution cycle, see Subsection 6.10.2. The enhanced selection mechanism suggested in Subsection 7.3.9 should provide a good utility to improve Detecting Uncolluding M(P)SPs who deploy large *Abuse Delay Periods*.

7.2.6 *Effects of Colluding MPSPs*

In our Simulation Process, we evaluated three main Colluding types as described in Subsection 5.8.7. The results of our Simulation reveal that the Popular MPSPs Colluding with a Large Pool of Unpopular MSPs type of colluding would have minimal effects on the integrity of the Trust Network due to its minimal attacking activities. Hence, this type of colluding should not pose a serious risk to the Trust Network's integrity despite its very Low *Detection Rates* by the Auditor.

The Second type of Colluding is the Popular Colluding MPSPs, which is the major threat to the Network in most of the cases. That is, by deploying either Strong Colluding or Weak Colluding Strategy, see Section 6.7, these Colluding MPSPs are able to compromise up to 10% of all the Users Credentials in the Trust Network with minimal *Detection Rates*. Deploying Aggressive Defensive Strategy's Flavours would significantly improve detecting these Colluding MPSPs but at the cost of significantly increasing the *Banned PSPs Rates* as well as the *Active Agents to Users Rate*.

Incorporating DGU in the Defensive Strategies' Flavours is effective in improving the *Popular Colluding Detection Rates*. However, given the fact that Strategies' Flavours deploying DGU would give the detected (M)PSPs the chance to avoid getting banned by agreeing to *Install DGU* means simply giving life to the Colluding MPSPs so they could continue their malicious activities by deploying the third type of Colluding Strategies, Advanced Colluding. Unfortunately, this Colluding Strategy is able to fool all our proposed Ranking Algorithms. Thankfully, this type of attack is applicable only when the DGU is utilized and, hence, we have time to improve our Algorithms until the DGU is ready for real-life deployment. Further, the proposed enhancement to the DGU Deployment Rules, see Subsection 7.3.3 could be the basis to improve our current Algorithms. The proposed Divide and Conquer Algorithm, see Subsection 7.3.9, could also improve how the *TGT* Approach Algorithms analyse the available *MGR* Records and how to create Testing Agents in a smarter way that lead to eventually improve the *Detection Rates* of all type of Colluding Attacks.

7.2.7 Effects of Popular SPs Committing Suicide Attack Combined with MSPs Registration Bombarding

This threat was not considered in our Simulation Model since we think it is not a realistic threat. That is because of the High cost of losing a popular MPSP, which takes a lot of efforts to publicise among real Users. Second, this threat would not work unless the utilized Defensive Strategy Flavour enables *Ignoring old G Ranks*. Among all the developed Strategies' Flavours, only one Strategy does enable this setting. Interestingly, it is the Aggressive Manual Flavour that we do not recommend deploying anyways.

Nevertheless, we will discuss here the possible consequences in case the Auditor did not detect the nature of the Suicide Attack and proceeded to ignore all the G Ranks where the suicider appears. As shown in Subsection 5.8.8, the $\tau_{MPSP-Halt}$ value would be reached by three triggers:

$$\tau_{MPSP-Halt} = \frac{C_{MR} * SP_{n-real}}{(C_{GR} * U_{GR}) + (C_{GR-newPII} * U_n) - (C_{MR} * SP_{GR-real})} \quad (7.1)$$

where SP_{n-real} is the current number of real SPs that Users are dealing with at a normal rate and $SP_{n-real} \geq 0$,

$SP_{GR-real}$ is the real SPs Growth Rate and $SP_{GR-real} \geq 0$.

Equation 7.1: Recovery Period after Detecting MPSP launching a combined Suicide and MSPs registration Bombarding Attacks
Considering the Soft Trust effects, $\tau_{MPSP-Halt}$

- $C_{MR} \approx \infty$: Such a C_{MR} value would be an indication that the deployed Defensive Strategy's Flavour is useless as it cannot detect M(P)SPs no matter

how many evidences it collects from the Network's Users. In such a case, the Trust Network could be either compromised even without a Suicide Attack or may have just reached an equilibrium status where there is almost no MPSPs Detection combined with minimal malicious activity.

- $SP_{nM} \approx \infty$: Such an SP_{nM} value means that while the MSPs Registration Bombarding attack by itself might be of little value for the ME, it would be very harmful for the Network when combined with a Suicide Attack as shown in Subsection 5.8.9. It is shown there that this combination could permanently paralyse the Auditor's Defensive Strategy. However, this should not be expected in most real-life scenarios. That is because Equation 5.12 does not consider the effects of the Soft Trust, see Subsection 2.8. For example, if a PSP, like Google, decides to launch the MSPs Registration Bombarding Attack, it would need to create a very large number of MSPs that needs to look legitimate and Trustworthy for Users. That means, simply recreating the same MPSP, Google in our example, by altering few details like the MSP title or background color will not fool the Users to Trust dealing with these new random entrants. i.e while Users would love to deal with Google, they would be very suspicious to deal with Google2, Google3, Google4 and so on specially that Google should refrain from relating itself to these dummy websites in order not to be blamed for any Abuse Cases coming from these websites. By considering the Soft Trust effect, Equation 5.12 could be shortened to Equation 7.1.
- $C_{GR} \approx 0$: Such a C_{GR} value would indicate a saturating Network where Users are not interested in it anymore. That could be due to lack of interesting (P)SPs or the lack of Trust on the Network. Since "Trust is hard to build but easy to lose [16]", it would be challenging to restore any lost Trust in the Network and, hence, a Suicide Attack on such a Network, without Counter-Defences like not *Ignoring Old G Ranks*, might be the death bullet for it.

7.2.8 *Effects of Introducing DGU*

Here we are not discussing a threat. Rather, we are trying to evaluate the benefits of introducing the hypothetical DGU unit to be utilized by the developed Defensive Strategies' Flavours, see Sections 4.4 and 5.6.3. During our Simulation Process, we have observed that the Defensive Strategies' Flavours employing the DGU unit have shown less dependency on Testing Agents and less *Banned PSPs Rates*, except for the Aggressive Strategy - Extreme DGU Flavour as shown in Subsection 6.6.8. In other words, the DGU unit has the desired effects of improving the accuracy of the Defensive Strategies. That is especially true knowing that the base Strategy Flavour that we used to optimize the deployment of the DGU unit on top of it prior to the DGU Refinement Stages is the Aggressive Strategy - Manual Flavour with significantly High Agents consumption and *Banned PSPs Rate*.

The main disadvantage of introducing the DGU unit is the fact that it gives the Colluding MPSPs second lives where they could deploy the Advanced Colluding Strategy. This type of Colluding cannot be detected, yet, by our proposed Ranking Algorithms, see Subsection 7.2.6. Nevertheless, the *Users' Trust Rate* would still be High while the *Compromised Credentials Rate* would still be Low since the Colluding Attackers would still be deploying a conservative Attacking Setting to avoid getting detected. We think further experimentation and research should yield better Ranking Algorithms that are capable of Detecting the Advanced Colluding Strategy. A first step would be to develop the proposed enhancements to the DGU Deployment Rules as described in Subsection 7.3.3.

The DGU would also have some interesting effects on light of our Malicious Density Theory, see Section 6.8. In that theory, we propose that a moderate density of M(P)SPs in the Network, not Low or High, would boost the detection of them even by applying a Random Aggressive Strategy Flavour. By employing the DGU unit, all (P)SPs who *Enabled Strict DGU* would be excluded from the calculations carried out by the Ranking Algorithms. Hence, those PSPs would be considered vanished in the perspective of the Malicious Density Theory leading to a High Density of MPSPs.

This High Density would push the Auditor into the Top Low Detection Region as shown in Figure 6.3. Nevertheless, since the employment of DGU pushes the Colluding Attackers to adopt the Advanced Colluding Strategy that our current Ranking Algorithms cannot recognise, those Colluding MPSPs could also be assumed to be vanished in the perspective of the Malicious Density Theory. That would lead to lowering down the Uncolluding MPSPs density to, perhaps, the Detection Threshold Region.

Finally, the effect of offering the Network's inhabitants the option to voluntarily *Install DGU* and/or *Enable Strict DGU* would mean constant *MGR* Records discarding as (P)SPs that appear in those Records would be eliminated after they adopt the DGU unit. This constant updates would lead to a temporary slowdown in the detection process. This temporary effect could turn into a permanent effect if the *Normal Nodes GR* continues High leading to a constant supply of new (P)SPs who keeps adopting the DGU after spending a while in the Network creating some *MGR* Records by their genuine interactions. This effect is illustrated in Figure 6.3.

7.3 Enhancements to the Developed Prototype

In this Section we introduce some enhancements to improve the performance of the proposed Auditor Model of Chapter 5 along with the optimized Defensive Strategies' Flavours of Chapter 6 as well as to enhance its immunity against the various types of threats that were introduced in Section 5.8 and Discussed in Section 7.2. We start with the idea of optimising the Defensive Strategies' Flavours release order for a wise dynamic Auditor. Further, we list some enhancements to improve different aspects of the developed Model such as the system's overall efficiency, Malicious Density Theory and DGU Deployment Rules.

7.3.1 Enhanced Defensive Strategies Release Order

Given the variety of the Defensive Strategies' Flavours that we have already developed, there must be a robust selection mechanism for the Auditor to pick the best

Flavour for the environment it is operating in. Real-life Networks are dynamic with Attackers continuously adapting their Counter-Attacking Strategies to fool the deployed Defensive Strategy's Flavour. Hence, a smart Auditor should keep an eye on its Health Gauges, see Subsection 5.6.4, to determine whether it is time to release a new Defensive Strategy.

In fact, our research on Defensive Algorithms in the Cyber Security Research Centre at Newcastle University shows that the order of the Defensive Algorithms' release does matter in prolonging the system's immunity against breaches even if it is theoretically possible to breach the system at certain time in the future [133]. In our preliminary experiment, we found that if there is a Strong Algorithm, usually more computationally expensive to deploy, that can be slightly modified by adjusting some of its variables or breaking it into smaller and easier to compromise Algorithms, then variations of that Algorithm could be released, each for a limited time, to prolong the system's immunity against compromise in comparison to deploying the original Strong Algorithm straight away. In addition, we found that deploying the toughest versions of the Strong Algorithm before the simpler ones would be more effective since this would disturb the normal learning curve where the learning Attacker would learn by breaking simpler defences before facing the more complicated defences.

In case of our Auditor, deploying a tough Aggressive Defensive Strategy's Flavour at the beginning does make perfect sense, see Section 6.7. That is because the Aggressive Strategy's Flavours demand more resources to operate, Testing Agents in particular. Hence, deploying an Aggressive Strategy in a small bootstrapping Network would cost much less than deploying it in a large Network. In addition, starting tough would mean that the bad residents, M(P)SPs, would be caught quickly leading to boosted *Users' Trust* and discouraging the MPSPs from trying to tamper with the Networks' integrity. Plus, passive malicious residents trying to launch Reverse Engineering Attacks, see Subsection 5.8.5, would reconstruct the toughest of the Defensive Strategies' Flavours as the default Auditor model. That would give the Auditor the advantage to relax its Defensive Strategy's Flavour while the MPSPs are not realising the change quickly enough, see Subsection 7.2.4. In fact, if the Auditor decides to be proactive by altering

its deployed Flavour frequently, it could take the lead in the optimisation game against the Counter-Attackers. That is, the Auditor could keep the passive malicious residents busy building a good approximate model of the Auditor's deployed Flavour without success, since it would be trying to model a Flavour that is not deployed anymore.

Example 7.1: Estimating the Density Region

- **Fact 1:** We learn from the Experimental Stages, see Section 6.6, that the MPSPs would cause about 90% of the generated Abuse in most of the cases.
- **Fact 2:** The SP_{nM-est} Health Gauge, see Subsection 5.6.4, gives an estimation of all the M(P)SPs in the Network by dividing the number of open Cases over the number of Cases it takes the Auditor to detect a single M(P)SP.
- **Fact 3:** From Fact 1 and Fact 2, $MPSPs \approx 90\% * SP_{nM-est}$.
- **Fact 4:** The most accurate Density Equation is $D_P = \frac{MPSPs}{PSPs+MPSPs}$, see Section 6.8.
- **Fact 5:** The Auditor could get the actual number of all the Service Providers, (M)(P)SPs, in the Network directly from the IDP.
- **Fact 6:** Knowing that all the Popular Service Providers, (M)PSPs, would be around 20% of all (M)(P)SPs per Pareto Rule [121], the denominator of Equation D_P can be estimated to be $PSPs + MPSPs \approx 20\% * (M)(P)SPs$.
- **Fact 7:** Using the MPSPs value from Fact 3 and the MPSPs + PSPs value from Fact 6, the D_P can be calculated to roughly determine the current Density Region.

In Section 6.9, we list few different Defensive Strategies' Flavours, each with its own strengths and weaknesses. As we stated above, a wise Auditor could start with one of the Aggressive Strategy's Flavours. Then, it should keep an eye on the performance of the Attackers just in case they have succeeded in figuring out the currently deployed Flavour and, hence, increasing their Abuse rates without getting detected. For that, the Auditor should implement the necessary Health Gauges of Subsection 5.6.4 to be

able to predict in which Density Detection Region he is operating in per the Malicious Density Theory of Section 6.8. Example 7.1 illustrates how this could work.

In addition, the Auditor should be able to sense the type of the dominant Counter-Attacks being launched against its deployed Defensive Flavour. That is, it should be able to relate the current Counter-Attacks to a known Counter-Attacking Strategy like Strong Colluding or Weak Colluding, see Section 6.7. That could be done by starting with, lets say, a Conservative Strategy Flavour to tackle Strong Colluding Attacks. Then, the Auditor could slightly alter its deployed Flavour to be more Weak Colluding Attack oriented, i.e. an Aggressive Strategy's Flavour. If the Auditor observes good improvements in the *Users' Trust Rate*, reduction in the *Compromised Credentials Rates*, and increased MPSPs Detection Rates, then that would signal that the current Counter-Attacking Strategy is more Weak Colluding Attack oriented. If the opposite observations are found, then the current Counter-Attacking Strategy is probably Strong Colluding Attack oriented. Also, the Auditor could try to relax its Defensive Flavour and monitor whether that would negatively affect the Trust metrics or not. If not, then maybe the Network is dominated with trivial attackers and, hence, the Auditor could relax the deployed Flavour for a while.

Knowing the current Density Region and the nature of the launched Counter-Attacking Strategy is essential to optimize the release order of the deployed Defensive Strategy Flavour. Table 6.14 compares the currently available Flavours and shows in which Density Region each of them is safe for deployment. The following rules provide more advanced guidance to aid the Auditor in its quest to deploy the most effective Flavour for the environment he is operating at:

- Deploy an Aggressive Strategy's Flavour when operating within the Density Threshold Region unless the Counter-Attacking Strategy Flavour is intensively Weak Colluding oriented and most (P)SPs have not *Installed DGU* yet. In that case, try to harness the Aggressive Strategy's Flavour with some Conservative settings. For example, limit the Weak Colluding analysis to the (P)SPs who did not *Install DGU*.

- Stop offering the possibility to voluntarily *Install DGU* when operating within the Density Threshold Region to boost the *MPSPs Detection Rates*. That is because the vDGU Conservative Flavour could push the Density Point toward the Top Low Detection Region, see Section 6.8.
- Deploy the Extreme DGU Aggressive Flavour when operating within the Density Threshold Region and most (P)SPs have already *Installed DGU*, to take advantage of the powerful *Detection Rates* of that Flavour in the Density Region where it is easy to detect most of the MPSPs. A big disadvantage of the Extreme DGU Aggressive Flavour is its High *Banned Innocent PSPs Rates*, but that is not a big concern if the Auditor already knows that most innocent PSPs have already *Installed DGU*. Alternatively, this powerful Flavour could be utilized to calculate the Ranks for only the (P)SPs who have already *Installed DGU*, to tackle the issue of the Advanced Colluding Strategy of Subsection 5.8.7.
- Deploy a Conservative Strategy's Flavour when operating within the Top/Bottom Low Detection Regions, to avoid banning innocent (P)SPs by an Aggressive Strategy's Flavour.
- Offer (P)SPs the possibility to voluntarily *Install DGU* when operating within the Top/Bottom Low Detection Regions. That is because the vDGU Conservative Flavour side effect of reducing the *MPSPs Detection Rates* would not be noticed within these Low Detection Regions. Rather, this step would be like an investment to gain better *MPSPs Detection Rates* when the Auditor moves to operate within the Density Threshold Region, see Subsection 7.2.8.
- Deploy the GPD Random Flavour when the dominant Counter-Attackers are trivial Uncolluding Attackers to reduce the demand for Testing Agents.
- Deploy the ST Conservative Flavour when the dominant Counter-Attackers are Uncolluding M(P)SPs. Alternatively, minimise the W_{TGT} to reduce the *Banned Innocent PSPs Rates*.

7.3.2 *Enhanced Malicious Density Theory*

The proposed Malicious Density Theory of Section 6.8 generally success in explaining most of the variations in the Detection Rates of the different Defensive Strategies' Flavours when operating at different environmental settings with different rates of Nodes Populations. Nevertheless, we have observed toward the end of our Experimental Process that each Flavour would have slightly different Density Regions Boundary Points, see Section 6.9. These observations would suggest that we could improve the current form of the theory by introducing separate Density Graphs for each of the different Flavours, or at least for each Defensive Strategy. To even improve the accuracy, those newly introduced Density Graphs should be three-dimensional. The third dimension would represent the type of the attack because some types of attacks would be easily detectable within a larger Threshold Density Zone, like the Uncolluding MPSPs Attack for example.

7.3.3 *Enhanced DGU Deployment Rules*

Employing the DGU unit in the Auditor's deployed Defensive Strategy's Flavour has many advantages in terms of improving the Auditor's detection accuracy. Nevertheless, that comes at the cost of encouraging the Colluding MPSPs to adopt the Advanced Colluding Strategy, giving them immunity against the Defensive Strategies as shown in Subsection 7.2.8. The Colluding MPSPs are able to adopt this type of Settings by misusing the Credentials sharing feature which is possible by the DGU. That is, a PSP_x who has already *Installed DGU* could legitimately share $Cred_x$ with SP_y who has not *Installed DGU* yet. When SP_y Abuses $Cred_x$, then the *MGR* Records would show, at least, the pair of PSP_x and SP_y in the suspicious Colluding chain, see Section 5.4. PSP_x would keep sharing with different PSPs, that have not *Installed DGU* yet, to Abuse Credentials in its behalf and, hence, the *TGT* Ranking Algorithms would not be able to distinguish a suspicious colluding chain to be responsible for the majority of the Abuse Cases.

To tackle this issue, the Auditor could **discard the option to *Install DGU***. That is, if any (P)SP decides to voluntarily adopt the DGU, it should *Enable Strict DGU*, see Subsection 5.6.3 straight away to completely eliminate the possibility of the Advanced Colluding Strategy Attacks. We anticipate that most (P)SPs would be reluctant to adopt this restrictive DGU mode at the early bootstrapping phase. Nevertheless, the more (P)SPs adopting it, the more the Malicious Density would increase, see Subsection 7.2.8. If this increased Density moves the Auditor from the Bottom Low Detection Zone to the Threshold Detection Zone, then the Detection Rates would be boosted leading to forcing the Colluding MPSPs to either *Enable Strict DGU* or leave the Network, which is a desired outcome. On the other hand, if the Auditor fails to deploy the correct Flavour to neutralise or eliminate most of the Colluding MPSPs in the Threshold Detection Region, see Section 6.8, the Malicious Density would continue to raise moving the Auditor to operate within the Top Low Detection Region where it would start to accuse innocent PSPs for Abuse Cases they did not commit. If innocent PSPs did not like the idea of forcing them to *Enable Strict DGU* and decided to leave, that may increase the *Banned PSPs Rates* to unacceptable levels. Further experimental studies are needed to evaluate the effects and the real-life (P)SPs altitude toward this approach.

7.3.4 Enhanced Users Sticky Policies Options

If the population of PSPs in real-life is reluctant to adopt the restrictive *Enable Strict DGU* as suggested in Subsection 7.3.3, then Users' pressure could help speeding up the process. That is, Users could assign their Credentials a new Sticky Policy called **Share with DGU**, see the simulated policies in Section 6.4. This policy basically means that the User is fine with sharing her Credentials with third-party partners provided that they are Trusted. That is, it forces the PSP acquiring those Credentials to act as if he has already *Enabled Strict DGU*, even if he has not yet, when handling those Credentials. The more Users assigning their Credentials with that policy, the more PSPs would be encouraged to adopt this more restrictive, but Trustworthy, mode.

7.3.5 Enhanced Ratings Publishing Mechanism

A wise Auditor should not publish any *TG* Ranks before waiting for a period of time $\tau_{publish}$ to prevent M(P)SPs from Reverse Engineering the Auditor's deployed Defensive Strategy's Flavour by associating an action they commit with a change in their *TG* Ranking, see Subsections 5.8.5 and 7.2.4. In other words, limiting the feedback M(P)SPs can get regarding the deployed Flavour. Plus, the Auditor could make $\tau_{publish}$ a variable value so that MEs would not be able to figure out when to look for the delayed Auditor's feedback to get published.

7.3.6 Enhanced Internal Ratings Records

In our developed Model, the *SPR* and *IIR* Records are introduced to aid the Auditor in selecting suspicious (P)SPs for investigations by the Testing Agents, see Subsections 5.6.1 and 5.6.2. This idea of classifying the different types of threats in specialised Records could be extended to cover the serious Threats described in the Threat Model, Section 5.8. That is, we could create specialised Records to log how likely a PSP_x to be involved in a certain attack type, like Uncolluding, Popular Colluding, or Advanced Colluding. For example, when the Auditor suspects that PSP_x is involved in a Colluding Popular Attack and, as a consequence, created a GT_x Agent who, eventually, got positive results confirming its suspicions, then it should add PSP_x to the Colluding Popular PSPs Record and increase his counter by 1. When another GT_y , testing a different G containing PSP_x , gets created with the same motivations, suspecting a Colluding Popular MPSPs Attack, and that GT_y got positive results confirming its suspicions, then PSP_x 's counter in the Colluding Popular PSPs Record should now become 2. Such Attacks' Records would help in improving the Testing Agents' efficiency as it would restrict creating Agents to the (P)SPs appearing in certain important Records only. The importance of the Records would be figured out if further Simulation Research takes place to test these ideas. Note that we have already implemented a beta version of such classification Records but the generated results are still unreliable, see Subsection 6.10.6.

7.3.7 *Enhanced User Reporting Handling*

To tackle the issue of genuine Users not reporting all their Abuse Cases, see Subsection 5.8.1, a good User interface design should be considered. Such a design should make reporting an Abuse a simple and straight forward task. Automating this task would be even better by deploying some filters that detects common spam patterns, for example. For Data breach cases, other than the spam scenario, a bot that search the web trying to find whether the Credentials of its owner are present somewhere in the Network without permission or not would be a valuable tool. If this bot could detect a Data breach, it could automatically report a breach Case without the need of consent from its owner.

Regarding the risk of malicious Users reporting inaccurate Cases, see 5.8.2, a User Reliability Index is something that should be considered by the Auditor. Such an Index would give weight for each reported Case based on how reliable a User would look like. This Index should consider factors like how real and Trustworthy a User is by checking its interactions history and whether he got friends, joined social Networks, got positive peer reviews, and got non-suspicious overall interactions pattern, see the Reputation Systems of Section 2.8. There are many challenges to design such an Index like maintaining Users' Privacy and the Index integrity. For example, the design must ensure that the collected Data are anonymised by a good anonymity factor to protect the Users' Privacy [25].

7.3.8 *Enhanced Testing Agents Performance*

The following is a list of techniques that would enhance how the Testing Agents Perform:

Agents Recycling: To improve the overall system efficiency, the Auditor should consider recycling expired STs by using them as seeds for new GT Agents. The same applies for expired GTs as they could be utilized to seed new GTs to test bigger Gs where the old Agent's G is a subgroup of the new G, see Subsection 5.4.1. Furthermore,

to improve the Auditor's efficiency, more STs and GTs could be created and utilized whenever more computational resources are available. However, this act should be planned with caution since if smart M(P)SPs observe a High volume of Testing Agents that are active at certain times of the day, perhaps with less traffic, they would then adjust their malicious activity to take place at the busier times of the day where less Testing Agents would be active.

Multitasking ST Agent 'with a set of non-testing Data': To confuse any Reverse Engineering Attack, see 5.8.5, an enhanced ST Agent could interact with more than one SP but with a different set of personal Data. This way, one ST Agent could be used to test several SPs at once.

Adapting the average User pattern: Since some M(P)SPs would refrain from Abusing Users that do not follow the average User interaction pattern, see 5.8.5, the Testing Agents should improve their interaction pattern by making it identical to the average User model. That could be done by analysing the reported Cases by Users to generate an average User model. This model should be embedded by all the created Testing Agents. Of course, the average User model assumed by the M(P)SPs would differ from that of the Auditor since the latter has access to a wider range of Users' logs. Hence, the Testing Agents' interaction settings should have some variability so that they could not form a recognizable pattern.

7.3.9 Enhanced Testing Agents Selection Algorithms

The following is a list of Selection Algorithms to aid the Auditor in deciding when and how to create new Testing Agents. Variations of these Algorithms could be deployed depending on the available computing resources and the release order plan, see Subsection 7.3.1.

Divide and Conquer GT Selector: If a GT_x is testing a large G_x gets an Abuse, the Auditor could decide to get bolder TGT values by using a Dividing Algorithm. Such Algorithm would divide G_x into smaller G_y and G_z . Then, the Auditor could create new GT Agents G_y and G_z for each of the newly generated Gs respectively.

The division could be random, by checking the G_x subsets that already appear in the *PIIL* Record even with a Low count, or by selecting the first few (P)SPs as they appear chronologically to tackle the possibility that these first few could have committed an Abuse but the User failed to report it, see Subsection 5.8.1.

Small PIILs Selector: A GT Selector that favours smaller PIILs could improve the efficiency and effectiveness of the Auditor. That is because small collations would be exposed in a quicker manner compared to Testing Larger Gs and then applying the Divide and Conquer Selector to it or keeping its less assertive *TGT* values. For that, it would be a good idea to select the top 20% of *PIILs* entries, in spirit of Pareto Principle [121], and then ordering the generated list based on G size. Then, start creating GT Agents for the smaller groups before heading for the larger ones.

Excluding Suspicious Colluding (P)SPs from ST Testing: Based on the proposed enhanced Attacks' Records of Subsection 7.3.6, the Auditor could decide to exclude suspicious (P)SPs from Selection for ST Agents Testing. That is because ST Agents cannot detect Colluding M(P)SPs anyways and, hence, it is wise not to waste the ST Agents resources on hopeless tasks.

Chapter 8. Conclusion and Future Directions

Before we conduct our research, there was very little work done to support Continuous Data Control by the existing Trust Management Frameworks. It is true there are many measures already existing to improve the Data Control Before Release for the Users' Credentials to be shared with third-parties. Nevertheless, the current Trust Frameworks do not support the vital Attribute of Data Control After Release to enforce the Trust rules on Users' Data after releasing them to the requesting third-parties, see Section 3.5. Meanwhile, our proposed Continuous Trust Framework Design achieves significant improvements to the Network's *Users' Trust* level in addition to significant reductions in the Users' *Compromised Credentials Rates* by introducing pioneering Defensive Strategies' Flavours to protect the Users' Credentials even after they are released.

Our approach to derive the final set of Defensive Strategies' Flavours was systematic, yet agile and dynamic. We started with carefully studying the literature to clearly understand the purpose of the Trust Frameworks and the needs of their main Stakeholders: the Users, Service Providers, and Legal Authorities. It was clear from the beginning that Trust is hard to build but very easy to destroy [16]. For that, our aim was to design and proof the concept of a Continuous Trust Management Framework that satisfies all its Stakeholders by supporting all the Trust Requirements we list in Section 2.5 including the vitally missing Continuous Data Control Attribute.

In our quest to satisfy the key Trust Frameworks Stakeholders, we propose the Continuous Trust Framework Design that is based on the best practices found in the literature. In addition, we introduce new building blocks to tackle the least covered, yet vital, Continuous Data Control Attribute, see Subsection 2.6.2. This Trust Attribute is essential for our proposal to be considered a Continuous Trust Management

Framework where the Trust it provides would start with the first instant of communication between any two parties and would last even after the transaction finishes. To prove the practicality of our proposed Continuous Framework, we implemented a minimal version of it on top of the OpenID Connect protocol, see Section 4.2.3. Our simple prototype shows how the Trust Stakeholders would interact in real-life and how the Auditor Unit that we introduce would aggregate Data access logs. We have further designed a hypothetical Data Governance Unit, DGU, that could be realised using Hard Trust Measures Level solutions like the TCG technologies [65]. The DGU unit would provide the ultimate Trust if it gets implemented and fully deployed by all the Trust Network's residents, See Section 4.4.

Given the fact that the DGU cannot be assumed to be available and widely adopted sometime soon, we introduced a set of Ranking Algorithms, Utilities, and Deployment Rules to aid in the design of a powerful Auditor that could make use of the aggregated Data access logs, see Chapter 5. Our Ranking Algorithms incorporate the innovative idea of creating artificial Testing Agents that could interact with suspicious Service Providers to detect their malicious acts. Through an extensive Simulation Process, we simulated and optimized how our Auditor would behave when operating in large Networks under different environmental situations, see Chapter 6. This Simulation process generated a set of different Defensive Strategies' Flavours that are suitable for deployment against different Counter-Attacking Strategies operating in different environmental settings. In fact, we came up with the Malicious Density Theory that predicts how effective the performance of the Auditor would be given the Density of malicious nodes present in the Network, see Section 6.8.

By the end of our Simulation Process, we carefully analysed our findings to build on the strengths and mitigate the weaknesses. We came up with a set of possible enhancements for the future as well as discussed the limited effects of the currently posed threats. In Section 7.1, we argue that the proposed Continuous Framework has proved its capability to achieve its ultimate aim: providing Continuous Trust for its Stakeholders throughout all their transactions phases. All in all, we believe that our contribution in this thesis is an important addition to the Trust computing literature

that would set the ground for a new era where the Trust Frameworks' Stakeholders would consider the Continuous Data Control an essential requirement rather than a science fiction!

8.1 Future Directions

In this Section, we list some of the future directions that we would love to take to improve our proposed Continuous Framework:

- **Implementing the DGU:** that is an important step that would solve many of the current challenges facing our proposed Continuous Framework. It would be challenging because it depends in its core on the TCG technologies, which are still novice. An alternative path would be to consider implementing the DGU based on the Blockchain theory that gained momentum in parallel to our research work, thanks to its infamous application: the Bitcoin. There are preliminary research work to utilize the Blockchain theory to enforce Continuous Data Control but there are still many concerns regarding the scalability of this new theory [134].
- **Incorporating the Simulation Model Algorithms in the Simple Prototype:** that is, we want to reuse the code we have already written to simulate the Auditor's Defensive Strategies' Flavours into our simple proof-of-concept prototype. This way, we will get a more advanced prototype that could be utilized to run pilot real-life experiments with real Users.
- **Implementing the proposed Enhancements:** that is, we want to implement the proposed enhancements of Section 7.3 first in the Simulation Model and then in the actual prototype to improve the current performance metrics. Particularly speaking, we wish to reduce the *Active Agents to Users Rate* to a very low value, something less than 10%, without affecting the *Malicious Providers Detection Rates*.

- **Running a real-life experiment based on a game model:** that is, developing and deploying an online simulation game representing the environment of the Continuous Framework. All the interactions would be done on one simulation server. Plus, there is no need for a large number of participants. A human player could start a game session by selecting one of the roles and then interacting with a system of automated entities (Users, SPs, Auditors, ... etc). It should be possible to have more than one human player though if there are many logged-in humans in the server. The player could setup the game session at the beginning so that she plays with a specific number of entities with specific percentages of traffic and malicious nodes Density or Counter-Attacking Strategies to test all the special cases that could arise in a real-life Continuous Framework. At the end of the session, the player would be given statistics showing how accurate the Auditor was during the simulation.
- **Implementing the Legal Conformance unit of the Continuous Framework:** this unit would be a unique addition to the Trust computing literature. It would basically store a Database of all the relevant legalisation covering how Data should be handled online. Then, it should screen any contract signed between Users and Service Providers to detect any policy that contradicts with one of the laws. For example, if a Service Provider asks for too many Credentials, he would be notified that this request is against the EU Data minimisation act.
- **Implementing the Negotiation unit of the Continuous Framework:** this unit would allow tailor made contracts to take place in our proposed **Continuous Framework**, which would enhance the experience of the Trust Network's Users and give them more satisfaction.
- **Extending the Auditor's protection to cover Users' Data:** this is a challenging task but could be possible by incorporating the UMA protocol, see Subsections 7.1.2 and 3.1.6.

Appendix A. Screenshots of the Implemented Prototype

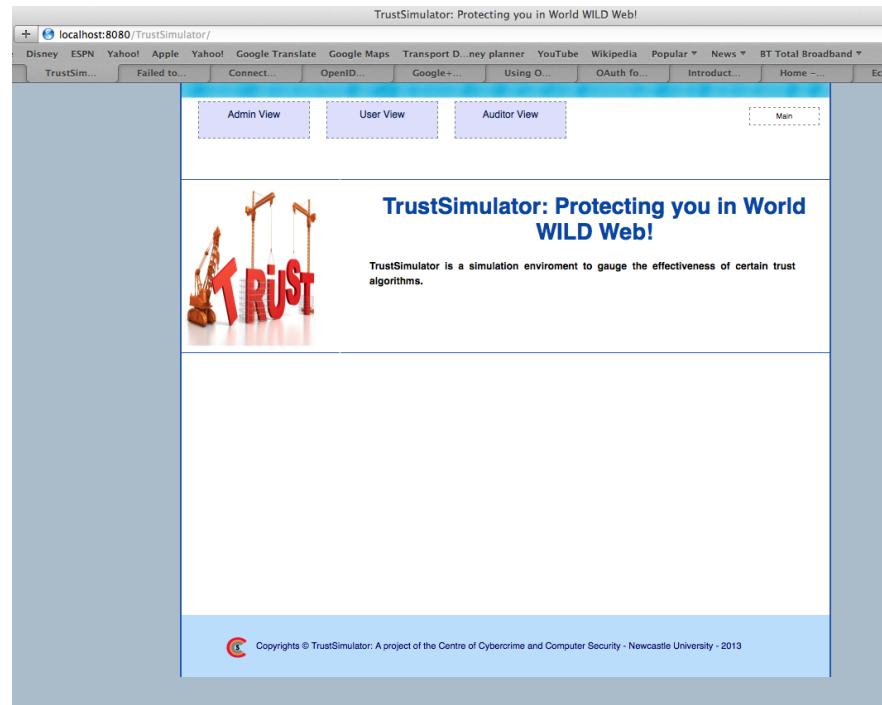


FIGURE A.1: Index Page

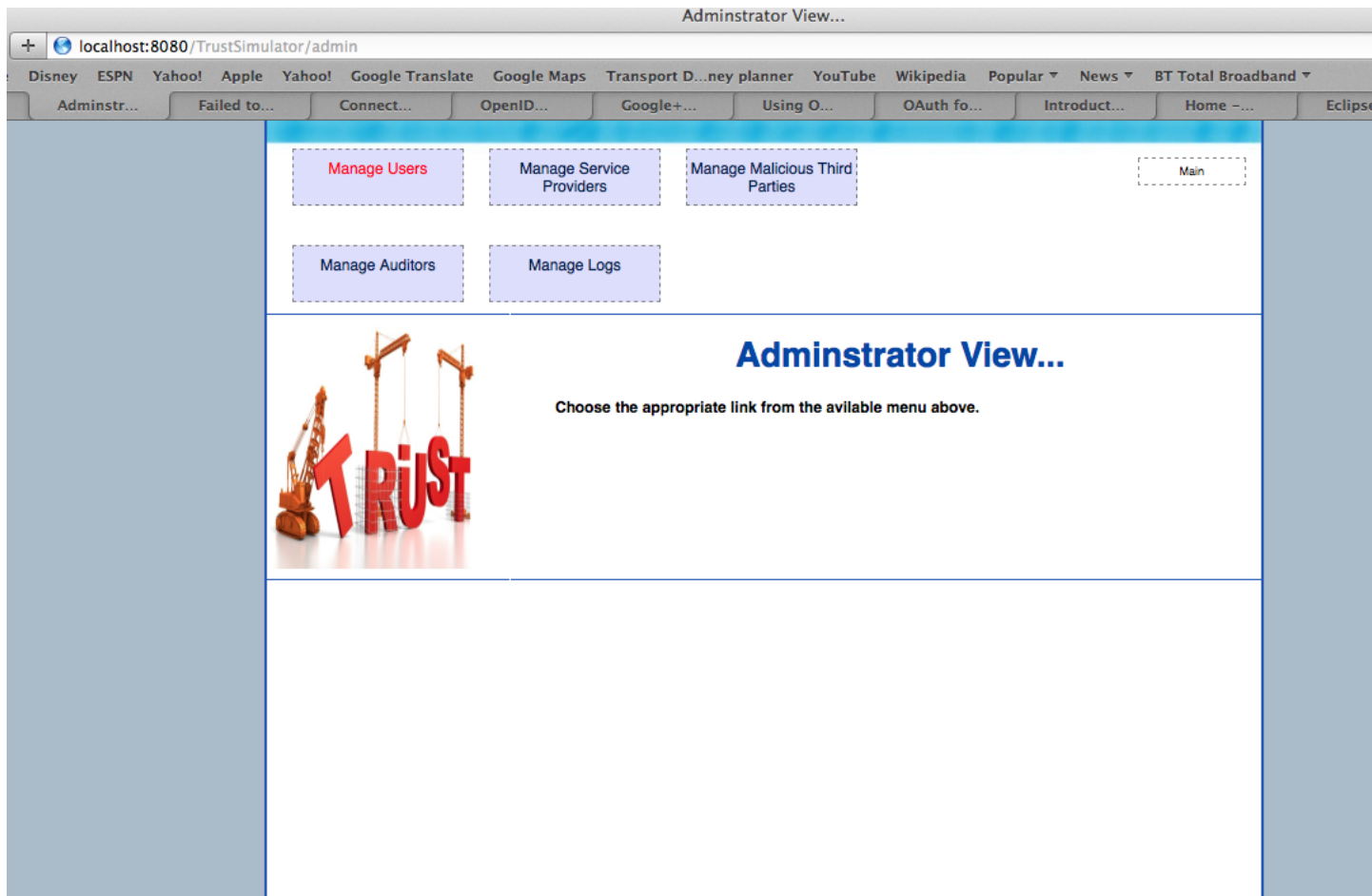


FIGURE A.2: Admin Page

The screenshot shows a web browser window with the URL `localhost:8080/TrustSimulator/deleteUser_1151`. The page title is "Manage Users...". Below the title, there is a message: "Below, you can Add / Edit / Delete Users:". The "Add a New User" section contains a green notification box that says "User Deleted Successfully". Below this, there are three input fields labeled "NAME", "PASSWORD", and "EMAIL". At the bottom of this section are two buttons: "Reset" and "Add User". The "Existing Users" section contains a table with the following data:

User ID	Name	Email	Created On	Last Modified	Actions
1	Ahmad	a2@test.co	2014-05-27 15:45:19.0	2014-05-27 15:45:36.0	Resume ; Edit ; Delete
2	Ahmad Alonnizi	onaizi01@hotmail.com	2014-03-10 17:05:31.0	2014-03-10 17:05:31.0	Resume ; Edit ; Delete
651	Ahmad 2	Ahmad2@test.com	2014-06-10 17:24:11.0	2014-06-10 17:24:11.0	Resume ; Edit ; Delete
1052	Ahmad 3	ahmad3@hotmail.com	2014-06-11 17:39:01.0	2014-06-11 17:39:01.0	Resume ; Edit ; Delete

FIGURE A.3: Manage Users Page

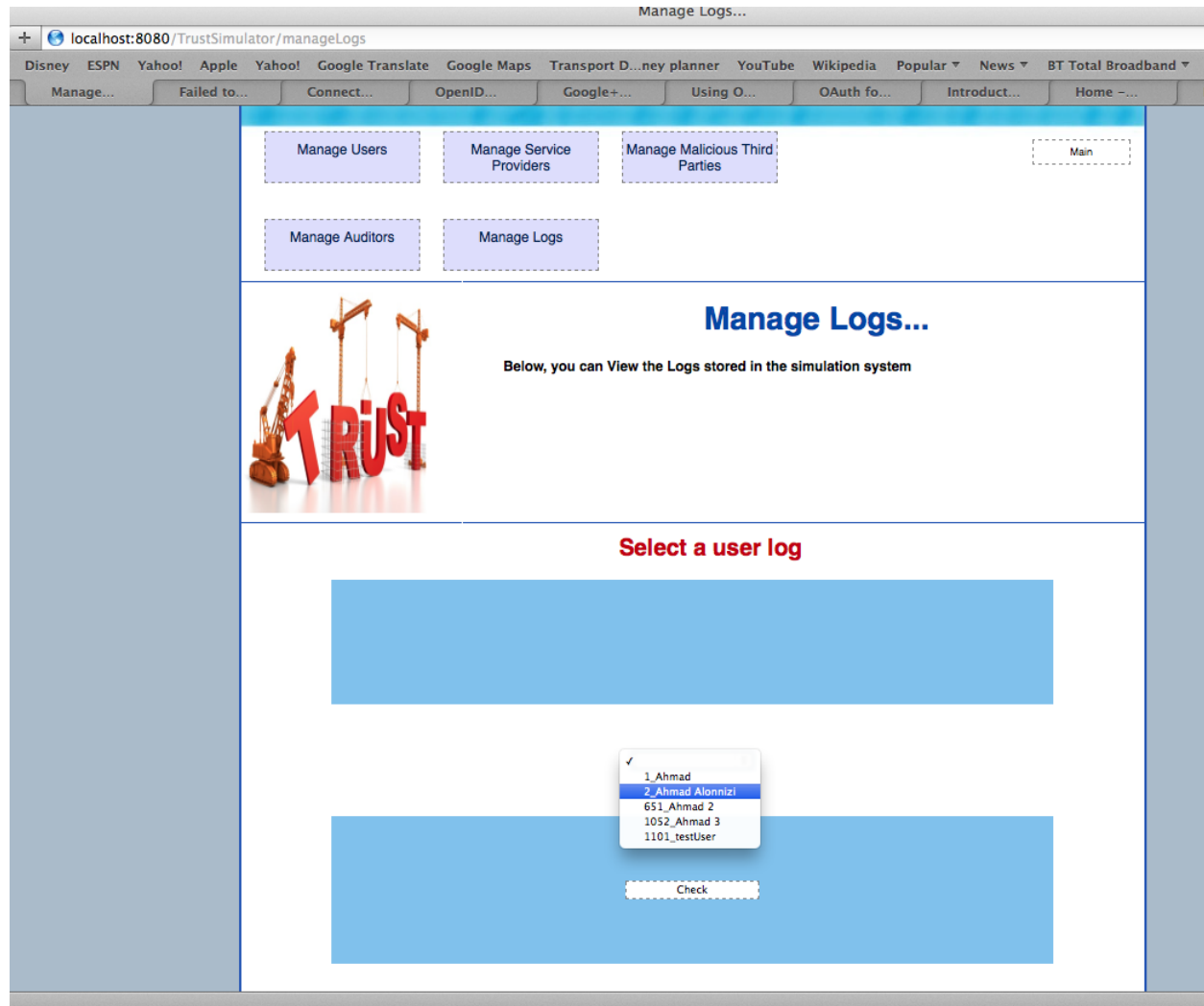


FIGURE A.4: Manage Logs Page1

localhost:8080/TrustSimulator/getUserLog

Disney ESPN Yahoo! Apple Yahoo! Google Translate Google Maps Transport D...ney planner YouTube Wikipedia Popular News BT Total Broadband

Manage... Failed to... Connect... OpenID... Google+... Using O... OAuth fo... Introd... Home -...

Manage Users Manage Service Providers Manage Malicious Third Parties Main

Manage Auditors Manage Logs

Manage Logs...

Below, you can View the Logs stored in the simulation system

Go back to show another Log

This is the Log for User: 1 whose Name is Ahmad

Time	Action	More Details
2014-05-27 16:24:46.0	SP 4 got user credentials	Expand
2014-05-27 16:24:46.0	SP 4 got user credentials	Expand
2014-05-27 16:24:46.0	MSP 12 got user credentials from SP 4	Expand
2014-05-27 16:24:46.0	User received Message 7 from SP_1@test.co.uk	Expand
2014-05-27 16:24:50.0	User received Message 9 from SPm@test.co.uk	Expand
2014-05-27 16:24:50.0	MSP 12 used user credentials	Expand
2014-05-27 16:24:50.0	User read Message 7	Expand
2014-05-27 16:24:53.0	User read Message 9	Expand
2014-05-27 16:24:57.0	AuditorCase 14 is created by Auditor 10 for a reported SPAM by User	Expand
2014-05-27 16:32:06.0	Auditor 10 updated AuditorCase 14	Expand
2014-05-27 17:23:03.0	User read Message 9	Expand

FIGURE A.5: Manage Logs Page2

Manage Logs...

localhost:8080/TrustSimulator/getExpandedLog_6

sney ESPN Yahoo! Apple Yahoo! Google Translate Google Maps Transport D...ney planner YouTube Wikipedia Popular News BT Total Broadband

Manage... Failed to... Connect... OpenID... Google+... Using O... OAuth fo... Introd... Home -...

Manage Users Manage Service Providers Manage Malicious Third Parties Main

Manage Auditors Manage Logs

Manage Logs...

Below, you can View the Logs stored in the simulation system

Go back to the User Log

Time	MSP Record ID	MSP ID	Leaked From SP ID	Data Obtained From User ID	Obtained Data Type	Obtained Data Value	Record Usage Count	Committed Action
2014-05-27 16:24:46.0	5	12	4	1	email	a2@test.co	0	save

FIGURE A.6: Manage Logs Page3

The screenshot shows a web browser window with the address bar displaying 'localhost:8080/TrustSimulator/resumeUserSession_651'. The browser's address bar and tabs are visible at the top. The main content area is divided into several sections:

- Header:** Contains two buttons: 'Edit Profile' and 'Inbox (2)'. There are also 'Main' and 'Logout User' buttons on the right side.
- Main Content:** Features a large image of a construction crane and the word 'TRUST' in large, red, 3D letters. To the right of the image, the text reads: 'User Simulation Session... In this simulation page, you can choose to edit your personal details or check your inbox using the menu tabs. Plus, you can start interacting with one of the service providers listed below:'
- Available Service Providers:** A table listing two service providers with their descriptions and offered services.

Service Provider	Description	Offered Service	Actions
SP 1		Gold	Get Service
SP 2		Diamond	Get Service

FIGURE A.7: User Simulation Session

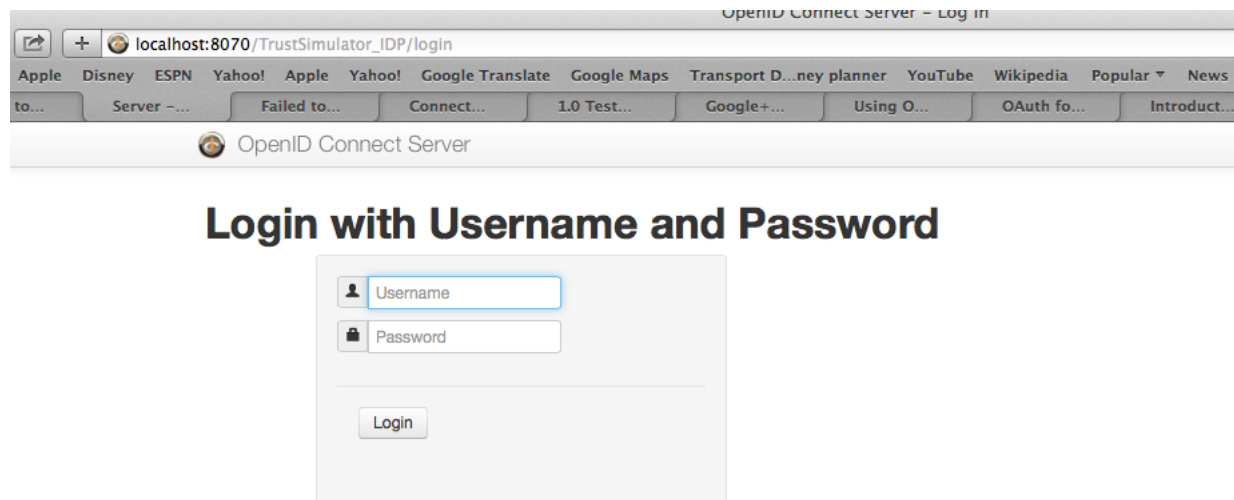


FIGURE A.8: User Simulation - Login IDP

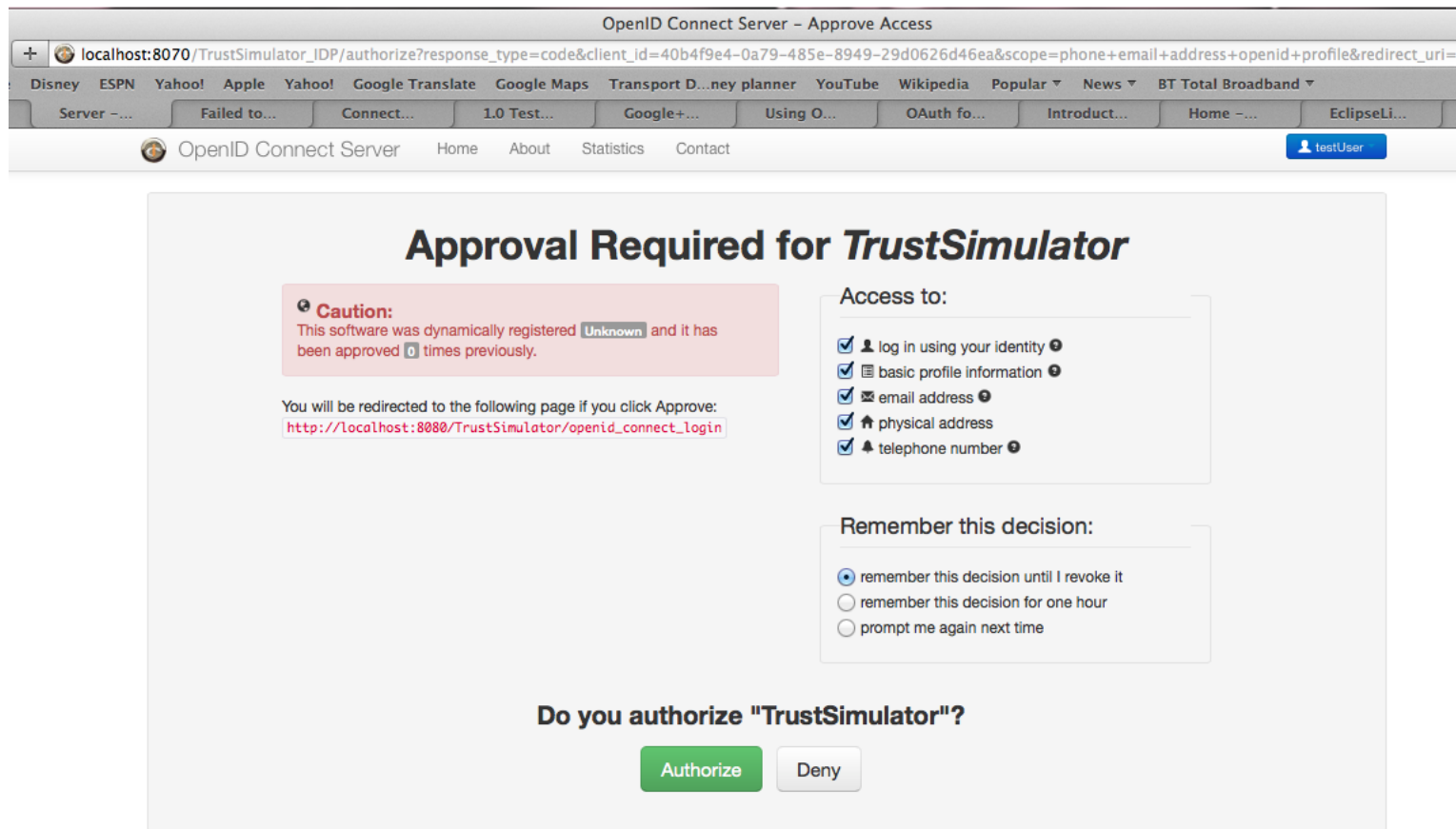


FIGURE A.9: User Simulation - Consent to IDP

You are about to send your following information:

NAME	Ahmad 2
EMAIL	Ahmad2@test.com

To the following Service Provider:

NAME	SP 2
DESCRIPTION	

In return for getting:

Diamond


 Copyrights © TrustSimulator: A project of the Centre of Cybercrime and Computer Security - Newcastle University - 2013

FIGURE A.10: User Simulation - Confirm Service Request

localhost:8080/TrustSimulator/inbox_651

Disney ESPN Yahoo! Apple Yahoo! Google Translate Google Maps Transport D...ney planner YouTube Wikipedia Popular News BT Total Broadband

User Sim... Failed to... Connect... OpenID... Google+... Using O... OAuth fo... Introd... Home -...

Edit Profile Inbox (4) Main Logout User

User Simulation Session...

In this simulation page, you can choose to edit your personal details or check your inbox using the menu tabs. Plus, you can start interacting with one of the service providers listed below:

Welcome to your Inbox

Sender Name	Title	Date	Status
MSP 1	MSP 1 - Product Offer!!	2014-06-20 22:50:05.0	unread
SP 2	SP 2 - Service Confirmation	2014-06-20 22:50:02.0	unread
MSP 1	MSP 1 - Product Offer!!	2014-06-11 15:14:15.0	unread
SP 1	SP 1 - Service Confirmation	2014-06-11 15:14:10.0	unread

FIGURE A.11: User Simulation - User Inbox

User Simulation Session...

localhost:8080/TrustSimulator/showMessage_1163

Disney ESPN Yahoo! Apple Yahoo! Google Translate Google Maps Transport D...ney planner YouTube Wikipedia Popular News BT Total Broadband

User Sim... Failed to... Connect... OpenID... Google+... Using O... OAuth fo... Introd... Home -... EclipseLi...

Edit Profile Inbox (3) Main Logout User



User Simulation Session...

In this simulation page, you can choose to edit your personal details or check your inbox using the menu tabs. Plus, you can start interacting with one of the service providers listed below:

[Back to your Inbox](#)

SENDER NAME	DATE	SENDER EMAIL	
MSP 1	2014-06-20 22:50:05.0	SPm@test.co.uk	
TITLE	MSP 1 - Product Offer!!	SPAM?	Report to an Auditor
BODY	<p>Congratulations!</p> <p>You Got an exclusive 75% discount to buy the best products FROM:</p> <p>MSP 1</p> <p>Do not miss this valuable offer, reply back NOW!!!</p>		

FIGURE A.12: User Simulation - MSP SPAM

localhost:8080/TrustSimulator/reportToAuditor_1163

Disney ESPN Yahoo! Apple Yahoo! Google Translate Google Maps Transport D...ney planner YouTube Wikipedia Popular News BT Total Broadband

User Sim... Failed to... Connect... OpenID... Google+... Using O... OAuth fo... Introd... Home -...

Edit Profile Inbox (3) Main Logout User

User Simulation Session...

In this simulation page, you can choose to edit your personal details or check your inbox using the menu tabs. Plus, you can start interacting with one of the service providers listed below:

Choose an Auditor

Auditor	Description	Actions
Auditor 1	NCL Lab	Report Spam

FIGURE A.13: User Simulation - Report SPAM - Choose Auditor

Auditor Simulation Session...

localhost:8080/TrustSimulator/resumeAuditorSession_10


Disney ESPN Yahoo! Apple Yahoo! Google Translate Google Maps Transport D...ney planner YouTube Wikipedia Popular News BT Total Broadband

Auditor S... Failed to... Connect... OpenID... Google+... Using O... OAuth fo... Introd... Home ... Ec

Edit Profile Main Logout Auditor

Auditor Simulation Session...

In this simulation page, you can choose to edit your personal details or check the current reported cases to you.



Currently Reported Cases

Case ID	User ID	Convicted SP ID	Date	Status	Actions
1167	651		2014-06-20 22:51:22.0	Open	View
14	1		2014-05-27 16:24:57.0	Closed	View


FIGURE A.14: Auditor Simulation

Auditor Simulation Session...

localhost:8080/TrustSimulator/getAuditorCase_1167

Disney ESPN Yahoo! Apple Yahoo! Google Translate Google Maps Transport D...ney planner YouTube Wikipedia Popular News BT Total Broadband

Auditor S... Failed to... Connect... OpenID... Google+... Using O... OAuth fo... Introd... Home -... Eclips



Auditor Simulation Session...

In this simulation page, you can choose to edit your personal details or check the current reported cases to you.

[Back to your Cases List](#)

REPORTING DATE		2014-06-20 22:51:22.0	
REPORTING USER NAME	Ahmad 2	REPORTING USER EMAIL	Ahmad2@test.com
INVESTIGATED DATA TYPE	email	INVESTIGATED DATA VALUE	Ahmad2@test.com
STATUS	Open	CONVICTED SP ID	
AUDITOR COMMENT		LAST MODIFIED	2014-06-20 22:51:22.0

Update Case		Update User	
SENDER NAME	MSP 1	SENDER EMAIL	SPm@test.co.uk
TITLE	MSP 1 - Product Offer!!	DATE	2014-06-20 22:50:05.0

BODY

Congratulations!

You Got an exclusive 75% discount to buy the best products FROM:

MSP 1

Do not miss this valuable offer, reply back NOW!!!

FIGURE A.15: Auditor Simulation - Case View

localhost:8080/TrustSimulator/showMessage_1170

Disney ESPN Yahoo! Apple Yahoo! Google Translate Google Maps Transport D...ney planner YouTube Wikipedia Popular News BT Total Broadband

User Sim... Failed to... Connect... OpenID... Google+... Using O... OAuth fo... Introd... Home -...

Back to your Inbox

SENDER NAME	Auditor 1	SENDER EMAIL	admin
DATE	2014-06-20 22:57:33.0		
TITLE	A SPAM Case Update from: Auditor 1	SPAM?	Report to an Auditor

This is an update from Auditor Auditor 1 regarding your SPAM report which you sent on: 2014-06-20 22:51:22.0

There is NO Convicted SP in your case yet

The current status of your report is: Closed

The current comment of the auditor is:No Clue!!

Below is the original message you reported:

SENDER NAME	MSP 1	DATE	2014-06-20 22:50:05.0
TITLE	MSP 1 - Product Offer!!		

Congratulations!

You Got an exclusive 75% discount to buy the best products FROM:

MSP 1

Do not miss this valuable offer, reply back NOW!!!

FIGURE A.16: User Simulation - View Auditor Update

Appendix B. Initial List of the Simulation’s Process Factors of Interest

This Appendix list all the factors of interest that we could come up with at the initial analysis of the simulated Auditor Model. The factors without Low and High values are the factors that were set constant and not chosen for the preliminary simulation experiments during this project. For more details, refer to Subsection 6.5.1.

Experiment Settings				
	Factor	Low	High	Notes
	Replicas	var		Depends on total of factors - 30 when all factors are constant
	Cycles	505		Higher values could cause out of memory errors

TABLE B.1: Factors of Experiment Settings

Environmental Factors				
	Factor	Low	High	Notes
	Network Size	$= 3 + iniUsers + iniPSPs + iniSPs + iniMPSPs + iniMSPs + iniME + iniME * (iniME_{MPSPs} + iniME_{MSPs})$ The 3 nodes are Super, IDP, and Auditor		
Grouped	iniUsers	50	130	ini prefix: initial number of nodes. GR suffix: percentage growth rate of nodes per simulation cycle.
	iniPSPs	8	20	
	iniSPs	16	40	
	UsersGR	0.05	0.2	
	SPsGR	0.05	0.1	
	PSPsGR	0.05	0.1	
Grouped	iniMPSPs	4	10	ini prefix: initial number of nodes. GR suffix: percentage growth rate of nodes per simulation cycle.
	iniMSPs	8	20	
	iniME_MPSPs	5	7	
	iniME_MSPs	8	12	
	MPSPsGR	0.05	0.1	
	MSPsGR	0.05	0.1	
	ME_MPSPsGR	0.05	0.1	
	ME_MSPsGR	0.05	0.1	

Environmental Factors				
	Factor	Low	High	Notes
	iniME	2		ME_1 Popular Colluding, ME_2 Unpopular Colluding.
	MEGR	0		No new MEs created dynamically
	ME-MaxMPSPs	15		
	ME-MaxMSPs	50		

TABLE B.2: Environmental Factors

Users' Controlled Factors. See Section 6.4 for more details.				
	Factor	Low	High	Notes
	Generate New Service Request Rate	50		
	Generate New Credential Rate	10		
	Strict Credential Prob	10		
	Ini Credential Type	0		0 = Strict, 1 = ShareWithTop, 2 = ShareWithAny
	Ignorance Rate	20	80	Probability a User would ignore reporting an Abuse
	Stop Using Credential After Spam	false		Users continue to use their Credentials even after receiving an Abuse
	$trust_o$	50		"Trust In Network" in Conf. Files.
	Trust In Network Dynamic	true		$trust_0$ for new Users should be dynamically adjusted to the average trust of current Users

TABLE B.3: Users' Controlled Factors

SPs' Controlled Factors. See Subsections 5.6.3 and 5.8.4 for more details.				
	Factor	Low	High	Notes
	new Partnership Prob	0.2		Probability of an SP making a partnership relation with another SP at a given simulation cycle.
	new Partnership Duration	50		The period, in cycles, of a newly created partnership relation.
	partner Sharing Prob	30		Probability of sharing a new Credential with a Partner
	Accept DGU Prob	0	40	Probability an SP voluntarily accepts to install DGU, if offered to by Auditor.

SPs' Controlled Factors. See Subsections 5.6.3 and 5.8.4 for more details.				
	Factor	Low	High	Notes
	Accept Compulsory DGU Prob	60		When SP detected, it should either accepts DGU or get banned
	Accept Strict DGU Prob	0	20	Probability an SP voluntarily accepts to enable strict DGU, if offered to by Auditor.
	Accept Compulsory Strict DGU Prob	40		When SP detected, it should either accepts DGU or get banned

TABLE B.4: SPs' Controlled Factors

Auditor's and IDP's Controlled Factors				
"SPAM" instead of "Abuse" in Conf. Files.				
	Factor	Low	High	Notes
	Update IDP Rankings Freq	1		how often Auditor updates IDP with latest ranks
	ini SP Rank	100		Initial rank of newly created SP at the bootstrapping cycle.
TG Ranking Settings. See Section 5.5 and Subsection 5.4.5 for more details.				
	$W_{TLR_{avg}}$	20	80	"TLRavg Weight" in Conf. Files.
	W_{TST}	20	80	"TST Weight" in Conf. Files
	W_{TGT}	20	80	"TGT Whole Weight" in Conf. Files.
	W_{sg}	20	80	"TGT Weight" in Conf. Files.
Grouped	W_{sc}	10	40	"TGT Colluding Weight" and "TGT Weak Colluding Weight" in Conf. Files.
	W_{wc}	40	10	
TLR_{avg} Settings. See Section 5.2 for more details.				
	Suspicious SP Rank	20	40	
	Suspicious SP Range	5		
	Suspicious SP Banning Rank	5	15	
Grouped	Sufficient TLRus PSP	5	20	Min TLRu ranks to ban (P)SP based on its TLR_{avg}
	Sufficient TLRus SP	2	10	
TST Settings. See Sections 5.3 and 5.7 for more details.				
	DeployST	false	true	False would turn off all the ST settings
	$\tau_{ST-idle}$	1		"ST Idle Time" in Conf. Files.
	$\tau_{ST-maxLife}$	20	80	"ST Max Life Time" in Conf. Files.
	ST Agent Status Post Abuse	3		1 = active, 2 = killed, 3 = killed and stop behaving as normal user

Auditor's and IDP's Controlled Factors				
"SPAM" instead of "Abuse" in Conf. Files.				
	Factor	Low	High	Notes
	ST Unimportant Credentials Pool	5		
Grouped with GT Report Abuse	ST Report Abuse	false	true	Reporting an Abuse like ordinary Users (using Unimportant Credentials)
	Top TLR_{avg} PCT-ST	20		To Aid ST Selectors
	Bottom TLR_{avg} PCT-ST			
	PSL PCT-ST			
	PSL ST Selector	false	true	ST Selectors. See TST Deployment Rules in 5.7.
	Suspicious Nodes ST Selector			
	Top TLR_{avg} ST Selector			
	Bottom TLR_{avg} ST Selector			
	IIR ST Selector			
TGT Settings. See Sections 5.4 and 5.7 for more details.				
	DeployGT	false	true	False would turn off all the GT settings
	$\tau_{GT-idle}$	1		"GT Idle Time" in Conf. Files.
	$\tau_{GT-maxLife}$	20	80	"GT Max Life Time" in Conf. Files.
	GT Agent Status Post Abuse	3		1 = active, 2 = killed, 3 = killed and stop behaving as normal user
	GT Unimportant Credentials Pool	5		
Grouped with ST Report Abuse	GT Report Abuse	false	true	Reporting an Abuse like ordinary Users (using Unimportant Credentials)
	PCR Threshold	2	7	
	PCR Weak Threshold	2	7	
	Ignore Old G Ranks	false	true	TGT ignores all Gs containing a banned SP
	GT Max SP Num	2	7	Max unpopular SPs in each G
	GT Max Size	2	7	Max G size
	PIIL PCT	20		
	PIIL GT Selector	false	true	GT Selectors. See TGT Deployment Rules in 5.7.
	PSL GT Selector			

Auditor's and IDP's Controlled Factors				
"SPAM" instead of "Abuse" in Conf. Files.				
	Factor	Low	High	Notes
	Suspicious Nodes GT Selector			
DGU Settings. See Subsection 5.6.3 for more details.				
	DeployDGU	false	true	False would turn off all the DGU settings
	Post DGU Installation Rank	20	80	
	Post Strict DGU Enable Rank	100		
Grouped	Offer DGU Prob	0	20	
	Offer Strict DGU Prob	0	10	

TABLE B.5: Auditor's and IDP's Controlled Factors

Attackers' Controlled Factors. See Subsections 5.8.6 and 5.8.7 for more details.				
"SPAM" instead of "Abuse" in Conf. Files.				
	Factor	Low	High	Notes
	Abuse Drop Strict Credential Prob	100		
Begin: Factors to be duplicated for M(P)SPs ∈ MEs				
	Abuse Delay MSP PCT	0		PCT of M(P)SPs deploying Delay attack only
	Abuse Delay MPSP PCT			
	Abuse Delay Period	5	50	Simple Attacking Strategy, see Subsection 5.8.6
	Abuse Bombarding MSP PCT	0		PCT of M(P)SPs deploying Bombarding attack only
	Abuse Bombarding MPSP PCT			
	Abuse Bombarding Period	5	50	Simple Attacking Strategy, see Subsection 5.8.6
	Abuse Drop MSP PCT	0		PCT of M(P)SPs deploying Drop User attack only
	Abuse Drop MPSP PCT			
	Abuse Drop User Rate	20	80	Simple Attacking Strategy, see Subsection 5.8.6
	Abuse Drop Credential Rate			

Attackers' Controlled Factors. See Subsections 5.8.6 and 5.8.7 for more details.				
"SPAM" instead of "Abuse" in Conf. Files.				
	Factor	Low	High	Notes
	Abuse Delay Bombarding MSP PCT Abuse Delay Bombarding MPSP PCT	0		PCT of M(P)SPs deploying Delay and Bombarding attacks only
	Abuse Delay Drop MSP PCT Abuse Delay Drop MPSP PCT	0		PCT of M(P)SPs deploying Delay and Drop attacks only
	Abuse Drop Bombarding MSP PCT Abuse Drop Bombarding MPSP PCT	0		PCT of M(P)SPs deploying Drop and Bombarding attacks only
	Abuse Delay Drop Bombarding MSP PCT Abuse Delay Drop Bombarding MPSP PCT	100		PCT of M(P)SPs deploying Delay, Drop and Bombarding attacks
End: Factors to be duplicated for M(P)SPs ∈ MEs				
	ME Colluding Unpopular PCT	50		Half MEs deploy large pool of MSPs colluding and the other half normal colluding. All would launch advanced colluding after installing DGU.
	$N_{toAbuse_U}$ $N_{toAbuse_C}$	1	5	"ME N MIN MPSPs to Abuse User" and "ME N MIN MPSPs to Abuse Credential" in Conf. Files.

TABLE B.6: Attackers' Controlled Factors

Monitored Outputs. See Section 6.1 for more details.	
"SPAM" instead of "Abuse" in Conf. Files.	
Output	Notes
All Outputs ending with <> are duplicated four times to record the same output during 4 different cycles: 125, 250, 375, and 500	
Trust In Network Users	Avg of all Users Trust by the end of simulation
Trust In Network Users 1-125	Avg Trust of Users created between cycles 1 and 125 by the end of simulation
Trust In Network Users 126-250	

Monitored Outputs. See Section 6.1 for more details. "SPAM" instead of "Abuse" in Conf. Files.	
Output	Notes
Trust In Network Users 251-375	
Trust In Network Users 376-500	
Trust In Network Users at <>	Avg Trust of all Users by <> cycle
Trust In Network Agents	Avg of all Testing Agents' Trust by the end of simulation
Trust In Network INI Agents	Avg of bootstrapping Agents' Trust at the end of simulation
Trust In Network Agents at <>	Avg Trust of all Agents by <> cycle
Active Agents to Users at <>	PCT of Agents among total Users
Avg Time To Resolve Case at <>	
PCT Undetected MPSPs at <>	
PCT Undetected MSPs at <>	
PCT Undetected Popular MPSPs at <>	MPSPs ∈ ME deploying normal colluding
PCT Undetected Popular MSPs at <>	
PCT Undetected Unpopular MPSPs at <>	MPSPs ∈ ME deploying colluding with a large pool of MSPs
Open cases at <>	
Open cases at <>	
PCT Compromised Share with Top Credentials at <>	
PCT Compromised Share with Any Credentials at <>	
PCT Abuse by MPSPs at <>	
PCT Abuse by MSPs at <>	
PCT Abuse by Popular Colluding ME at <>	
PCT Abuse by Unpopular Colluding ME at <>	
Avg Abuse to Ban MPSP at <>	
Avg Abuse to Ban MSP at <>	
Avg Abuse to Ban Popular Colluding MPSP at <>	
Avg Abuse to Ban Popular Colluding MSP at <>	
Avg Abuse to Ban Unpopular Colluding MPSP at <>	
Avg Abuse to Ban Unpopular Colluding MSP at <>	
Avg Abuse to Ban MPSP at <>	Avg Num of Abuses an MPSP sends out before getting banned
Avg Abuse to Ban MSP at <>	
Avg Abuse to Ban Popular Colluding MPSP at <>	

Monitored Outputs. See Section 6.1 for more details. "SPAM" instead of "Abuse" in Conf. Files.	
Output	Notes
Avg Abuse to Ban Popular Colluding MSP at <>	
Avg Abuse to Ban Unpopular Colluding MPSP at <>	
Avg Abuse to Ban Unpopular Colluding MSP at <>	
PCT PSPs Banned Guilty at <>	PCT of Innocent PSPs banned because they were thought Guilty
PCT SPs Banned Guilty at <>	
PCT PSPs Banned DGU at <>	PCT of Innocent PSPs banned because they refused installing DGU
PCT SPs Banned DGU at <>	
PCT PSPs Banned Strict DGU at <>	PCT of Innocent PSPs banned because they refused enabling strict DGU
PCT SPs Banned Strict DGU at <>	
The following outputs are duplicated for SPs, MPSPs, MSPs, Popular Colluding MPSPs, Popular Colluding MSPs, Unpopular Colluding MPSPs, and Unpopular Colluding MSPs	
PCT PSPs Installed DGU at <>	
PCT PSPs Enabled Strict DGU at <>	
(Beta Implementation) the following outputs are duplicated for SPs, MPSPs, MSPs, Pop- ular Colluding MPSPs, Popular Colluding MSPs, Unpopular Colluding MPSPs, and Un- popular Colluding MSPs	
PCT PSPs Detected by TLRavg at <>	
PCT PSPs Detected by TST at <>	
PCT PSPs Detected by TGT_{sg} at <>	
PCT PSPs Detected by TGT_{sc} at <>	
PCT PSPs Detected by Agent at <>	
PCT PSPs Detected by Total Rank at <>	

TABLE B.7: Monitored Outputs

Appendix C. Detailed Milestones Data of the Simulation Stages

In Section 6.6, the Experimental Stages of our Simulation Process were briefly described. In this Appendix, we list the detailed analysis of the Milestones that made up those Stages. In other words, here we list the intermediate results we have obtained before arriving to the conclusions we have listed in Sections ??, 6.8, and 6.14.

C.1 The Milestones of Stage 1: The no Auditor Case Followed by Initial Optimisation

Milestone 1.1: No Auditor Case

- **Simulated Factors:** *Ignorance Rate, Normal Users & (P)SPs population, Malicious (P)SPs population.*
- **Factors with most Significant Effects:** The *Ignorance Rate* has significant positive effect on *Users Trust* followed by the *Normal Nodes Population*. The *Malicious Nodes Population* has significant, but milder, negative effects on the *Users Trust*. The Interaction of High *Ignorance Rate* and High *Normal Nodes Population* as well as the interaction of High *Ignorance Rate* with Low *Malicious Nodes Population* have mild positive effects on the *Users Trust*.
- **Attack Characteristics:** Trivial attacking algorithm where the M(P)SPs would simply abuse any Credential they acquire.
- **Main Attackers:** *Uncolluding MPSPs* and *Popular Colluding MPSPs* are equally the main threats causing together around 90% of the network Abuse Rate.
- **Trust Status:** Users Trust is between 20% when the *Ignorance Rate* is low and 65% when it is high.
- **Final Settings for the following Milestone:** *Ignorance Rate* is set to low assuming a Users' population that is serious about its privacy. *Normal and Malicious Nodes Populations* are set to low to avoid running out of memory during the simulation process, a technical limitation which we discuss in Section 6.10.

Milestone 1.2: TLRavg Approach is Introduced and Optimised

- **Simulated Factors:** TLRavg Settings of Table 6.5 and the Attackers Factors of Table 6.6.
- **Factors with most Significant Effects:** The *Abuse Delay Period* has a significant negative effect on the *MPSPs Detection Rate* as well as a mild negative effect on the *Banned Innocent PSPs Rate*. The *Suspicious SP Rank* has a mild positive effects on the *MPSPs Detection Rate* but also with a mild negative effect on the *Banned Innocent PSPs Rate*. The *Sufficient TLRus SP* has a mild negative effect on the *MSPs Detection Rate* as well as a mild positive effect on the *Banned Innocent SPs Rate*.
- **Attack Characteristics:** The Attackers were manually set not to collude because it is not needed to fool this algorithm. This Milestone was repeated twice. The first time we forced the attacking settings to be trivial: short *Abuse Delay & Bombarding Periods* between 1 and 5, and small *Users & Credentials Drop Rates* between 5 and 20. The second time we forced more realistic ranges as described in Table 6.6. In both cases, the automatic optimiser for the attackers did not require setting their attacking factors High because it is possible to compromise the deployed defenses with less complicated attacks. Increasing the attacks complexity would be an overkill that would reduce the percentage of the compromised Credentials the attackers would gain.
- **Main Attackers:** The *Uncolluding MPSPs* are the main threat generating over than 55% of the total Abuse followed by the *Popular Colluding MPSPs* generating over than 30% of the total Abuse.
- **Defense Characteristics:** Poor performance even in the trivial attacks case. Given the poor detection figures, the TLRavg factors were optimised at modest values to avoid banning innocent (P)SPs. An interesting observation is that during high *Abuse Bombarding* attacks, the Auditor tends to optimise the *Suspicious SP Rank* and *Suspicious SP Banning Rank* at high levels to eliminate those bombarding entities ASAP. Nevertheless, the High *Abuse Delay Period* attack setting forced the Auditor to slow down banning (M)PSPs, by increasing the *Sufficient (P)SP TLRus* factor, to reduce the *Banning Innocent PSPs Rate*.
- **Trust Status:** In both attacking scenarios, the *Users' Trust* reaches 20% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached more than 50% and 65% respectively.
- **Auditor Performance:** Poor *Detection Rates* reaching 0% in most cases even with trivial attacking settings and the *Banned Innocent PSPs Rate* reached over 2%.
- **Final Settings for the following Milestone:** *Suspicious SP Rank* = 32, *Suspicious SP Banning Rank* = 10, *Sufficient TLRus PSP* = 20, *Sufficient TLRus SP* = 5, *Abuse Bombarding Period* = 28 (this attacking factor was fixed due to its low impact).

Milestone 1.3: TST Approach is Introduced and Optimised

- **Simulated Factors:** W_{TLRavg} , W_{TST} , and TST Settings of Table 6.5, except for the *ST Report Abuse* factor, and the Attackers Factors of Table 6.6 except for the *Abuse Bombarding Period*.
- **Factors with most Significant Effects:** The $\tau_{ST-maxLife}$ has a significant negative effect on the *Active Agents to Users Rate*. The *Abuse Delay Period* has significant negative effects on the *M(P)SPs Detection Rates*. The interaction of *Abuse Delay Period* with the $\tau_{ST-maxLife}$ where $\tau_{ST-maxLife}$ is larger than *Abuse Delay Period*, would have mild positive effects on the *M(P)SPs Detection Rates*. The $N_{toAbuse_C}$ with significant positive effects on the *Users' Trust* while the $N_{toAbuse_U}$ would have

mild effects on that response. The *Users and Credentials Drop Rates* have mild positive effects on the *Users Trust* as well as mild negative effects on the *Uncolluding M(P)SPS Detection Rates*. The *Top/Bottom TLRavg ST Selectors* have huge positive effects on the *Active Agents to Users Rate* as well as mild positive effects on the *Uncolluding MSPs Detection Rates*. The *PSL ST Selector* has significant positive effects on the *Active Agents to Users Rate*.

- **Attack Characteristics:** The Attackers were forced to deploy more sophisticated colluding attacks to avoid banning their valuable MPSPs. That is, setting *Users and Credentials Drop Rates* = 80, $N_{toAbuse_U} = 5$ but keeping $N_{toAbuse_C} = 1$. Hence, almost all the attackers were not detected.
- **Main Attackers:** The *Uncolluding MPSPs* are the main threat generating over than 45% of the total Abuse followed by both the *Popular Colluding MPSPs* and *Uncolluding MSPs* where each of them generating around 25% of the total Abuse.
- **Defense Characteristics:** The Auditor started the optimisation process with great *Uncolluding M(P)SPs Detection Rates* causing the attacking entities to deploy some forms of colluding attacks. When it comes to the ST selectors, turning on either of the *Top/Bottom TLRavg ST Selectors* would lead to selecting more than 20% of the the whole SPs population leading to a sharp increase in the number of Testing Agents. This increase is undesirable in our optimisation process, see 6.1. Therefore, the Auditor is inclined to turn on the *PSL ST Selector* which selects the most important category of attackers: the MPSPs. Given the strong colluding attacks, those settings would fail at improving the *Detection Rates* causing the Auditor to switch on only one of the aggressive *Top/Bottom TLRavg ST Selectors* since the M(P)SPs' deployed simple strategies could make their rankings bounce between the Top and the Bottom of the TLRavg record. However, the more complicated the attacks settings get, particularly the colluding settings, the more useless the TST Approach gets at detecting such attacks leading the Auditor to turn on all it's ST selectors in hope of improving the detection rates. It should be noted that the more complicated the Attacks get, the less Abuse that would be generated leading to better *Users Trust*.
- **Trust Status:** The *Users Trust* reached 90% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 2.5% and 0.5% respectively.
- **Auditor Performance:** Poor *Popular/Unpopular Colluding Detection Rates* reaching 0% while the *Uncoluding MPSPs* and *Uncolluding MSPs* Detection Rates reached around 30% and 10% respectively. The *Banned Innocent PSPs Rate* reached about 0.9% and the *Active Agents to Users Rate* reached about 80%.
- **Final Settings for the following Milestone:** $\tau_{ST-maxLife} = 57$, $W_{TLRavg} = 44$, $W_{TST} = 56$, all TST selectors switched on.

Milestone 1.4: TGT Approach is Introduced and Optimised

- **Simulated Factors:** TG and TGT Settings of Table 6.5 and the Attackers Factors of Table 6.6 except for the *Abuse Bombarding Period*.
- **Factors with most Significant Effects:** The $\tau_{GT-maxLife}$ has mild negative effects on the *Active Agents to Users Rate*. The *PIIL GT Selector* has a mild positive effect on the *Active Agents to Users Rate*. The $N_{toAbuse_C}$ has a significant positive effect on the *Users' Trust*, a mild positive effect on the *Active Agents to Users Rate*, and a significant negative effect on the *Colluding Popular MPSPs*

Detection Rate. The $N_{toAbuse_U}$ has a mild positive effect on the *Users Trust* as well as a significant negative effect on the *Colluding Popular MPSPs Detection Rate*. The interaction of High $N_{toAbuse_U}$ and High $N_{toAbuse_C}$ has a significant positive effect on the *Users Trust* as well as significant negative effect on the *Colluding MPSPs Detection Rate*. The *Users/Credentials Drop Rates* have mild negative effects on the *Users Trust*, *Active Agents to Users Rate*, *Uncolluding M(P)SPS Detection Rates*, and *Colluding Popular MPSPs Detection Rate*.

- **Attack Characteristics:** Similar to the previous Milestone 1.3.
- **Main Attackers:** The *Popular Colluding MPSPs* are the main threat generating around 75% of the total Abuse followed by the *Uncolluding MPSPs* generating around 15% of the total Abuse.
- **Defense Characteristics:** As the attacks got more complicated, the internal TG and TGT Ranking Weights did not have any significant effect. When it comes to the GT Selectors, we noted that at low level attacks, the *PIIL GT Selector* would mistakenly cause more innocent (P)SPs to be banned while that effect would almost vanish at high level attacks. That could be because of the large volume of logs generated at low level attacks with few PSPs on them leading to the false illusion of colluding relations. Hence, at low level attacks, the Auditor decided to switch off all the *GT Selectors*. However, once the attack gets a bit tougher, it turned on all the Selectors but the *PIIL GT Selector* to tackle the current threats. At the highest attack level, all the Selectors would be of little impact at detecting very advanced colluding settings. Therefore, the Auditor preferred to turn all the *GT Selectors* off in order to save on the cost of creating new Agents. However, we believe that since the Auditor is indifferent to switching on or off the GT Selectors during high levels attack, it would have been better to switch on the *PIIL GT Selector*, see Stage 6.6.4. Generally speaking, if the Auditor can detect the nature of the attacks the network is currently receiving, see 5.6.4, it should try to minimise the $W_{TGT_{sg}}$ and/or turn off the *PIIL GT Selector* during low level attacks to avoid mistakenly banning innocent (P)SPs.
- **Trust Status:** The *Users Trust* reached 97% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 8% and 3% respectively. The reason why *Share with Top Credentials* are having higher probability of being abused is the fact that they are obtained and shared only by the top (M)PSPs which increases the chance of being acquired and abused by an MPSP. That could be a problem with our simulation setting because we put a percentage of MPSPs that equal or sometimes outnumber the PSPs population, i.e. we are simulating the pessimistic scenario, see 6.10.
- **Auditor Performance:** Poor *Popular/Unpopular Colluding Detection Rates* reaching 0% while the *Uncolluding MPSPs* and *Uncolluding MSPs* Detection Rates reached around 30% and 10% respectively. The *Banned Innocent PSPs Rate* was 0% and the *Active Agents to Users Rate* reached about 85%.
- **Final Settings for the following Milestone:** $\tau_{GT-maxLife} = 55$, $PCR\ Threshold = 5$, $PCR\ Weak\ Threshold = 5$, $Ignore\ Old\ G\ Ranks = true$, $GT\ Max\ SP = 5$, $GT\ Max\ Size = 5$, $W_{TLRAvg} = 22$, $W_{TST} = 28$, $W_{TGT} = 50$, $W_{TGT_{sg}} = 50$, $W_{TGT_{sc}} = 25$, $W_{TGT_{wc}} = 25$, all TGT selectors switched on except the *PIIL GT Selector*.

C.2 The Milestones of Stage 2: 1st Optimisation Refinement of Attacks and Auditorial Settings

Milestone 2.1: Testing the importance of TST and optimising the two groups of Attackers (Uncolluding and Colluding)

- **Simulated Factors:** *Deploy ST*, *Top TLRavg ST Selector*, *IIR ST Selector*, and the Attackers Factors of Table 6.6 except for the *Abuse Bombarding Period*.
- **Factors with most Significant Effects:** The *Deploy ST* factor has a huge negative effect on the *Total Agents to Users Rate*, 80% rise in that rate when *Deploy ST* is turned on. *Deploy ST* also has mild positive effects on the *Uncolluding M(P)SPs Detection Rates*. When the internal ST selectors were examined, it turned out that the *Top TLRavg ST Selector* is the main influencer of the *Deploy ST*'s observed effects, noting that the *Bottom TLRavg ST Selector* is believed to have the same influence if it is turned on. The $N_{toAbuse_C}$ has a positive significant effect on the *User's Trust* and, together with $N_{toAbuse_U}$, significant negative effects on the *Colluding Popular MPSPs Detection Rate*. The *Users/Credentials Drop Rates* have mild positive effects on the *Users Trust* as well as mild negative effects on the *Uncolluding M(P)SPs Detection Rates*.
- **Attack Characteristics:** In this Milestone, we separated the Attackers into two groups: *Uncolluding Attackers* and *Colluding Attackers*. For the *Uncolluding Attackers*, the toughest simple strategies settings were selected. Interestingly, it seems they prefer if the *Colluding Attackers* are not setting the highest levels of colluding, i.e. they wish if the Auditor focus its' detection efforts on the other competing group of Attackers. That could be because if the population of Attackers falls down to a certain limit, it would be very hard for the Auditor to detect any further MPSPs because the available logs and Algorithms would be insufficient to distinguish the tiny population of MPSPs among the Normal innocent population, see Section 6.8. For the *Colluding Attackers*, they choose the opposite settings. i.e. lowest values for the simple strategies settings and the highest values for both the $N_{toAbuse_U}$ and $N_{toAbuse_C}$.
- **Main Attackers:** The *Popular Colluding MPSPs* are the main threat generating around 70% of the total Abuse followed by the *Uncolluding MPSPs* generating slightly less than 20% of the total Abuse.
- **Defense Characteristics:** Switching off *Deploy ST* lowered down the *Active Agents to Users* from around 80% to less than 5%, bearing in mind that the TGT most aggressive selector, *PIIL GT Selector* is also turned off since the last stage. That desired result came with some side effects. First, it is advantageous for the Auditor to turn on the TST Approach during low level attacks to quickly eliminate the *Uncolluding Attackers* or trivial *Colluding Attackers*. Second, even during high level attacks, the *Uncolluding Attackers Detection Rate* would fall from 20% to 6% if the *Deploy ST* is switched off. That would lead to adding more noise to the open logs since the TGT G's would get filled with Undetected *Uncolluding Attackers* which would be thought to be colluding. Moreover, the increase in the number of open logs along with the increase in their sizes would lead to accusing many innocent PSPs which would be thought to be colluding. This situation would lead the *Colluding MPSPs* to relax their attacking simple strategies in order to compromise more Credentials and to increase the wealth of reported cases to make it even more challenging for the Auditor to make correct acquisitions. Ideally speaking, if the Auditor can detect the nature of the deployed attacks against it, it should turn on *Deploy ST* during low level attacks and turn it off during high level attacks. Even when we tried to check whether

deploying only the *Top TLRavg ST Selector* and/or the *IIR ST Selector* to tackle this dilemma, we observed that the *Top TLRavg ST Selector* is the main influencer of the TST Approach generated responses. However, the *IIR ST Selector* showed acceptable *Uncolluding MPSPs Detection Rate*, not *Uncolluding MSPs Detection Rate*, during low level attacks with only 40% of *Active Agents to Users Rate*. Interestingly, we noted that when only the *Top TLRavg ST Selector* is switched on, the colluding attack would get tougher. We initially thought the reason might be just random or it could be that during high level attacks, the *IIR ST Selector* would initially detect more *Uncolluding MPSPs* leaving more room for the *Colluding MPSPs* to hide among a larger population of innocent PSPs appearing in the reported logs. Whereas without that selector, the TGT Approach would have to analyse more logs to detect colluding groups, thanks to the *Ignore Old G Rank* factor which gives the benefit of the doubt for both innocent and guilty (P)SPs by ignoring to consider all Gs containing a banned (P)SP in the TGT calculations. Alternatively, we thought the reason might be simply that the tougher the colluding attack gets, the easier it would be for the *Uncolluding MPSPs* to attack since they would go unnoticed in the long pile of unresolved cases generated by the *Colluding MPSPs*. In other words, there is a potential conflict of interest between the Colluding and the Uncolluding attackers affecting the optimisation process and, hence, untying the two categories of attackers was decided from this point on, see 6.10. In fact, toward the end of the whole simulation process, we figured out that the most likely reason for the tougher colluding attacks when we turn off the *IIR ST Selector* would be explained by the Malicious Density Theory which we developed during this simulation process, see 6.8 for more details.

- **Trust Status:** The *Users Trust* reached 97% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 7% and 4% respectively.
- **Auditor Performance:** Poor *Popular/Unpopular Colluding Detection Rates* reaching 1% while the *Uncolluding MPSPs* and *Uncolluding MSPs Detection Rates* reached around 10% and 12% respectively. The *Banned Innocent PSPs Rate* was 0% and the *Active Agents to Users Rate* reached about 85%.
- **Final Settings for the following Milestone:** *Deploy ST* = true, all TST selectors turned off except for the *Top TLRavg ST Selector*, *Uncolluding Abuse Delay Period* = 50, *Uncolluding Abuse User/Credential Drop Rates* = 80, *Colluding Abuse Delay Period* = 5, *Colluding Abuse User/Credential Drop Rates* = 20, $N_{toAbuse_U} = 5$, and $N_{toAbuse_C} = 5$.

Milestone 2.2: Optimising TLRavg and TST Approaches against Uncolluding Attacks

- **Simulated Factors:** W_{TLRavg} , W_{TST} , TLRavg and TST Settings of Table 6.5, except for the *ST Report Abuse* factor.
- **Factors with most Significant Effects:** Similar to Milestone 1.3, the $\tau_{ST-maxLife}$ has significant negative effects on the *Active Agents to Users Rate* and mild positive effects on the *Uncolluding M(P)SPs Detection Rates*. This confirms our point in Milestone 1.3 that $\tau_{ST-maxLife}$ must be larger than *Abuse Delay Period* in order to capture *Uncolluding M(P)SPs*. The *Top/Bottom TLRavg ST Selectors* have significant positive effects on the *Active Agents to Users Rate*.
- **Attack Characteristics:** Fixed since Milestone 2.1.
- **Main Attackers:** The *Uncolluding MPSPs* are the main threat generating around 48% of the total Abuse followed by the *Popular Colluding MPSPs* generating around 38% of the total Abuse.

- **Defense Characteristics:** The Auditor performance is similar to Milestone 1.3. However, since the attacks are now optimised at more complicated level compared to the initial optimisation settings in Milestone 1.3, the Auditor seems to get less confident at detecting the smart M(P)SPs deploying tough colluding as well as simple attacking strategies. Hence, we observed the tendency to minimise the W_{TST} to its minimum in favour of increasing the W_{TLRavg} that we know since Milestone 1.2 is a weak and inaccurate detection algorithm. In other words, the Auditor is tempted to deploy random detecting algorithms during very complicated attacks. Of course that would lead to an increase in the *Banned Innocent PSPs Rate* and, hence, the Auditor tries to minimise this side effect by increasing the *Suspicious SP Rank* and *Banning SP Rank* as well as the *Sufficient TLRUs to Ban PSPs* to their utmost levels.
- **Trust Status:** Slightly better than Milestone 1.3. The *Users Trust* reached 99% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 2.5% and 1% respectively.
- **Auditor Performance:** The *Popular Colluding MPSPs*, *Uncolluding MPSPs* and *Uncolluding MSPs* Detection Rates reached around 20%, 18%, and 10% respectively. The *Banned Innocent PSPs Rate* reached about 0.5% and the *Active Agents to Users Rate* reached about 60%.
- **Final Settings for the following Milestone:** *Suspicious SP Rank* = 40, *Suspicious SP Banning Rank* = 15, *Sufficient TLRus PSP* = 9, *Sufficient TLRus SP* = 10, $\tau_{ST-maxLife}$ = 80, W_{TLRavg} = 30, W_{TST} = 10, all TST selectors switched on except for *PSL ST Selector* and *IIR ST Selector*.

Milestone 2.3: Optimising TGT Approach along with the “Deploy ST” option against the Colluding Attacks

- **Simulated Factors:** TG and TGT Settings of Table 6.5, and *Deploy ST*.
- **Factors with most Significant Effects:** Similar to Milestone 1.4, the $\tau_{GT-maxLife}$ with mild negative effects on the *Active Agents to Users Rate*. The *Deploy ST* factor with a huge positive effect on the *Active Agents to Users Rate*. The *PIIL GT Selector* also has a significant positive effect on the *Active Agents to Users Rate*. Interestingly, the *Deploy ST* factor has a troublesome relation with the *PIIL GT Selector*. When *Deploy ST* is turned on while *PIIL GT Selector* is turned off, the *User’s Trust* would increase by about 2% and the *Banned Innocent PSPs Rate* would fall by 1%. But when we switch off *Deploy ST* and switch on *PIIL GT Selector* we notice that the *Uncolluding MPSPs Detection Rate* falls from 20% to 4% and the *Colluding Popular MPSPs Detection Rate* would increase from 40% to 45% compared to the Rates we could get when both Selectors are switched on together. We also noted that turning on *Deploy ST* would always generate an 80% figure of *Active Agents to Users Rate*. Nevertheless, switching on the *PIIL GT Selector* while *Deploy ST* is off would generate a rate of only 40% *Active Agents to Users*.
- **Attack Characteristics:** Fixed since Milestone 2.1.
- **Main Attackers:** The *Uncolluding MPSPs* are the main threat generating around 58% of the total Abuse followed by the *Popular Colluding MPSPs* generating around 28% of the total Abuse.
- **Defense Characteristics:** Similar to Milestone 1.4, the internal TG and TGT Ranking Weights did not have any significant effect. Nevertheless, we noticed that the Auditor is tempted to set the W_{TGT} , W_{wc} , and W_{sc} to high levels in order to get more aggressive acquisitions during the current high level

attack. However, the high cost of increasing the *Banning Innocent PSPs Rate* held the Auditor back and forced it to keep $W_{TGT} = 80\%$ but also increasing W_{sg} to 80% instead of giving more weight to the more aggressive GT Detectors: W_{sc} and W_{wc} . That is a compromise to get some of the aggressive detection powers the GT Detectors while trying to reduce the *Banned Innocent PSPs Rate*. When it comes to the GT Selectors and the *Deploy ST* factor, we noted similar results to Milestone 2.1. That is, turning on the TST Approach would always generate the 80% figure *Active Agents to Users Rate* while turning on the *PIIL GT Selector* while *Deploy ST* is off would generate a rate of 40% *Active Agents to Users*. Since the TST Approach is crucial to detect *Uncolluding MPSPs* while the *PIIL GT Selector* is also crucial to detect *Colluding MPSPs*, the Auditor decided to keep both Selectors on. Finally, due to the highly complicated attacks launched against the network, the Auditor is tempted to act as a random Detector. Hence, it decided to reduce both the *pcrThreshold* and *pcrWeakThreshold* to their minimum values which would, in turn, make more aggressive banning decisions. As a result, the *Banned Innocent PSPs Rate* rose from 0.05% to 1.8% despite the Auditor's decision to slow down the banning decisions by setting the *Ignore Old G Ranks = true*. i.e. ignoring all those ranks where a banned (P)SP appears.

- **Trust Status:** The *Users Trust* reached 99% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 1% and 3% respectively.
- **Auditor Performance:** The *Popular Colluding MPSPs*, *Uncolluding MPSPs*, *Uncolluding MSPs* and *Unpopular Colluding MPSPs* Detection Rates reached around 30%, 25%, 10%, and 5% respectively. The *Banned Innocent PSPs Rate* reached about 1.8% and the *Active Agents to Users Rate* reached about 80%.
- **Final Settings for the following Milestone:** $\tau_{GT-maxLife} = 80$, *PCR Threshold* = 2, *PCR Weak Threshold* = 2, *Ignore Old G Ranks* = true, *GT Max SP* = 2, *GT Max Size* = 2, $W_{TLRAvg} = 16$, $W_{TST} = 4$, $W_{TGT} = 80$, $W_{TGT_{sg}} = 80$, $W_{TGT_{sc}} = 16$, $W_{TGT_{wc}} = 4$, all TGT selectors switched on except the *PSL GT Selector*.

C.3 The Milestones of Stage 3: 2nd Optimisation Refinement of Attacks and Auditorial Settings

Milestone 3.1: Optimising the two groups of Attackers (Uncolluding and Colluding)

- **Simulated Factors:** The Attackers Factors of Table 6.6.
- **Factors with most Significant Effects:** The $N_{toAbuse_C}$ factor has significant positive effects on the *User's Trust* and the *Active Agents to Users Rate*. Both the $N_{toAbuse_C}$ and the $N_{toAbuse_U}$ factors have significant negative effects on the *Colluding MPSPs Detection Rates* either by themselves or by the Interaction of their High values. The *Uncolluding Users/Credentials Abuse Drop Rate* has a mild negative effects on the *Colluding MPSPs Detection Rates*. The *Uncolluding Abuse Delay Period* has mild positive effect on the *Active Agents to Users Rate*.
- **Attack Characteristics:** For the *Uncolluding Attackers*, the toughest simple strategies settings were selected except for the *Abuse Bombarding Period*. Actually, the interaction of setting both the *Users/Credentials Drop Rates* high would yield the ultimate result at reducing the *Uncolluding M(P)SPs Detection Rates*. That could be explained by the fact that low *Users Drop Rate* would mean many

Users reporting Cases containing the same (P)SP while low *Credential Drop Rate* would mean that a small group of Users, which are not dropped by a given *Uncolluding MPSP* would keep reporting Cases against that same *Uncolluding MPSP* leading the Auditor to catch it at the end of the day. For the *Colluding Attackers*, the most influential factors to reduce the *Colluding Detection Rates* were both the $N_{toAbuse_U}$ and $N_{toAbuse_C}$ followed by, with less extent, the *Users/Credentials Drop Rates* belonging to *Colluding MPSPs*. We noticed that setting the *Users/Credentials Drop Rates* belonging to the *Colluding MPSPs* low would increase the *Compromised Credentials Rates* as expected. Nevertheless, setting $N_{toAbuse_C}$ low while setting $N_{toAbuse_U}$ high would give the best combination to maximize the *Compromised Credentials Rates* without risking an increase in the *Colluding Detection Rates*. That might be because these settings would relax the colluding policy, enabling the Attackers to get as much Credentials as possible without risking an increase in the *Colluding MPSPs Detection Rates*. That is because setting $N_{toAbuse_U} = 5$ while reducing $N_{toAbuse_C}$ from 5 to 1 would technically mean launching *Weak Colluding Attacks* which are tricky to catch and easier to launch. That is because this attack requires the targeted Credential to be dealt with a less number of *Colluding MPSPs*, see 5.4.4 and 5.8.7.

- **Main Attackers:** The *Popular Colluding MPSPs* are the main threat generating over 60% of the total Abuse followed by the *Uncolluding MPSPs* generating over than 25% of the total Abuse.
- **Defense Characteristics:** In the previous stage, it was obvious that the Auditorial Defensive Strategy Flavour Flavour was bold and random at detecting and banning suspicious SPs. For that, when the *Colluding Attackers* decided at this Milestone to focus more on *Weak Colluding Attacks*, more Cases are now generated making it trickier for the Auditor to process and make banning decisions based on them even with the bold and random detection policy of the previous stage. The *Colluding Attackers* decision to increase their *Colluding Abuse User/Credential Drop Rates* to higher levels made it even more trickier for the Auditor to detect them. Another interesting point that we have noted is the positive relation between the triviality of the attacks and the *Banned Innocent PSPs Rate*. That might be because the current Defence Strategy Flavour Flavour is more random than evidence based, in order to function better under uncertain environment. That would come at the cost of rising risk of trivial attacks, launched in purpose, to increase the *Banned Innocent PSPs Rate* which would, in turn, increase the *Malicious MPSPs Density* for the benefit of the Attackers, See 6.8.
- **Trust Status:** The *Users Trust* reached about 95% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 9% and 5% respectively.
- **Auditor Performance:** Poor *Unpopular Colluding Detection Rate* and *Popular Colluding Detection Rate* reaching around 2% and 4% respectively while the *Uncolluding MPSPs* and *Uncolluding MSPs* Detection Rates reached around 15% and 5% respectively. The *Banned Innocent PSPs Rate* reached about 0.6% and the *Active Agents to Users Rate* reached about 85%.
- **Final Settings for the following Milestone:** *Uncolluding Abuse Delay Period* = 50, *Uncolluding Abuse Bombarding Period* = 5, *Uncolluding Abuse User/Credntial Drop Rates* = 80, *Colluding Abuse Delay Period* = 48, *Colluding Abuse Bombarding Period* = 5, *Colluding Abuse User Drop Rate* = 60, *Colluding Abuse Credential Drop Rate* = 76, $N_{toAbuse_U} = 5$, and $N_{toAbuse_C} = 1$.

Milestone 3.2: Optimising TLRavg and TST Approaches against Uncolluding Attacks

- **Simulated Factors:** W_{TLRavg} , W_{TST} , $TLRavg$ and TST Settings of Table 6.5, except for the *ST Report Abuse* and $\tau_{ST-maxLife}$ factors.
- **Factors with most Significant Effects:** The *Top/Bottom TLRavg ST Selectors* have significant positive effects on the *Active Agents to Users Rate*. The interactions of *Top/Bottom TLRavg ST Selectors* as well as the *Suspicious ST Selector* High values would have mild positive effects on the *Active Agents to Users Rate*.
- **Attack Characteristics:** Fixed since Milestone 3.1.
- **Main Attackers:** The *Popular Colluding MPSPs* are the main threat generating around 65% of the total Abuse followed by the *Uncolluding MPSPs* generating around 25% of the total Abuse.
- **Defense Characteristics:** The Auditor performance is similar to Milestone 2.2. However, we observed in this Milestone that the Auditor is confused whether to increase its randomness bold Defensive Strategy Flavour Flavour in order to detect more *Colluding Attackers*, mostly by pure luck, or to decrease that randomness behaviour in order to detect more *Uncolluding Attackers*. Detecting the latter is now a more systematic job, thanks to the lowered *Abuse Bombarding Period*. Since the main target of this step was to detect *Uncolluding Attackers*, the settings boosting their detection were preferred over those needed to detect the *Colluding Attackers*. That is, the *Suspicious Rank* was reduced while the *Sufficient TLRUs to Ban PSPs* was increased. Interestingly, the W_{TST} was decreased and the *Top/Bottom TLRavg ST Selectors* were turned off while the *PSL/Suspicious ST Selectors* were turned on. In other words, the Auditor decided to operate the *TST Approach* at its minimum capacity. That was not expected because it reduced the *Uncolluding MPSPs Detection Rates*. This unexpected optimisation could be explained by the gain of reducing the *Active Agents to Users Rate* as well as the slight increase in the *Users Trust*.
- **Trust Status:** The *Users Trust* reached around 95% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 9% and 5% respectively.
- **Auditor Performance:** The *Uncolluding MPSPs*, *Popular Colluding MPSPs*, and *Uncolluding MPSPs* Detection Rates reached around 19%, 2%, and 2% respectively. The *Banned Innocent PSPs Rate* reached about 0.25% and the *Active Agents to Users Rate* reached about 80%.
- **Final Settings for the following Milestone:** *Suspicious SP Rank* = 20, *Suspicious SP Banning Rank* = 15, *Sufficient TLRus PSP* = 20, *Sufficient TLRus SP* = 2, W_{TLRavg} = 16, W_{TST} = 4, all *TST selectors* switched off except for *Suspicious ST Selector* and *IIR ST Selector*.

Milestone 3.3: Optimising TGT Approach along with the “Deploy ST” option against the Colluding Attacks

- **Simulated Factors:** TG and TGT Settings of Table 6.5, except for $\tau_{GT-maxLife}$, and *Deploy ST*.
- **Factors with most Significant Effects:** The *Deploy ST* is now less influencing than *PILL GT Selector*. The latter now has mild positive effects on *Users Trust*, *Active Agents to Users Rate* as well as mild negative effects on the *Colluding Popular MPSPs Detection Rate*. Moreover, we noted that turning on *PILL GT Selector* would always generate around a figure of 80% *Active Agents to Users Rate*. Nevertheless, switching on the *Deploy ST* while *PILL GT Selector* is off would generate a rate of around 40% *Active Agents to Users*. That is just the opposite of the case in the previous refinement Stage, see Milestone 2.3.

- **Attack Characteristics:** Fixed since Milestone 3.1.
- **Main Attackers:** The *Popular Colluding MPSPs* are the main threat generating around 65% of the total Abuse followed by the *Uncolluding MPSPs* generating around 20% of the total Abuse.
- **Defense Characteristics:** Unlike what we were expecting, the Auditor choose almost the opposite settings by setting *PCR Threshold* to 3, *PCR Weak Threshold* to 7, and *Ignore Old G Ranks* to true. The reason could be the fact that there is no significant effect for those factors on the *Colluding/Uncolluding Detection Rates* and, hence, the Auditor preferred to reduce the *Active Agents to Users Rate*. In comparison to Milestone 3.3, *GT Max Size* has increased to accommodate the increasing population of *Colluding MPSPs* generated by the current Weak Colluding Attack while both the *PCR Threshold* and *PCR Weak Threshold* have increased to reduce the randomness behaviour of the Auditor. When it comes to the *Agents Selectors*, we found that despite the positive effect of the *PIIL GT Selector* on *Users Trust*, it has negative effects on the *Active Agents to Users Rate* and *Uncolluding/Colluding MPSPs Detection Rates*. The reason might be the higher volume of logs and the slower detection caused by the optimised setting *PCR Weak Threshold* to low, setting increasing *Ignoring old G Ranks* to true, and increasing *GT Max Size*. That combination of setting would cause generating weaker ranks which, in turn, restricts the Auditor's ability to quickly ban Suspected SPs leading to the described negative impacts. The sense of increased *Users Trust* might be because of the reduced activities caused by the intermediate decline in *Users Trust*, meaning that deploying the *PIIL GT Selector* under such conditions would mean it would function well for a limited capacity of traffic before breaking down until the traffic reduces again to the permissible limits. In other words, this combination of settings does not scale. As a result, the Auditor decided to turn off the *PIIL GT Selector* as well as the *Suspicious GT Selector*, which has milder effects than the *PIIL GT Selector*.
- **Trust Status:** The *Users Trust* reached 92% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 7% and 5% respectively.
- **Auditor Performance:** The *Popular Colluding MPSPs*, *Uncolluding MPSPs*, and *Unpopular Colluding MPSPs* Detection Rates reached around 14%, 11%, and 2% respectively. The *Banned Innocent PSPs Rate* reached about 0.25% and the *Active Agents to Users Rate* reached about 36%.
- **Final Settings for the following Milestone:** *PCR Threshold* = 3, *PCR Weak Threshold* = 7, *Ignore Old G Ranks* = true, *GT Max SP* = 2, *GT Max Size* = 7, $W_{TLR_{avg}} = 16$, $W_{TST} = 4$, $W_{TGT} = 80$, $W_{TGT_{sg}} = 25$, $W_{TGT_{sc}} = 60$, $W_{TGT_{wc}} = 15$, all TGT selectors switched off except the *PSL GT Selector*.

C.4 The Milestones of Stage 4: Comparing the Performance of Manual and Automatic Optimisation Settings

Manual Vs. Automatic Defensive Strategies Settings			
Factor	Manual	Automatic	Notes
TG Ranking Settings. See Section 5.5 and Subsection 5.4.5 for more details.			
$W_{TLR_{avg}}$	4	16	Under high level Attacks, $W_{TLR_{avg}}$ becomes just a random factor that should not be relied upon for fair banning decisions.

Manual Vs. Automatic Defensive Strategies Settings			
Factor	Manual	Automatic	Notes
W_{TST}	16	4	W_{TST} can prove if an <i>Uncolluding M(P)SP</i> is acting maliciously, but cant prove the opposite and, hence, is of little importance.
W_{TGT}	80		
W_{sg}	20	25	W_{sg} is of little importance when $N_{toAbuse_C}$ is low, which is highly expected from optimised Attackers.
W_{sc}	30	60	Same as the above note.
W_{wc}	50	15	Opposite the above note.
<i>TLR_{avg}</i> Settings. See Section 5.2 for more details.			
Suspicious SP Rank	30	20	Useful to detect <i>Uncolluding MPSPs</i> quickly before Open Cases pile up. However, it should not have been altered for the Manual Strategy Flavour Flavour since it has no effect without enabling either the <i>Suspicious ST Selector</i> or <i>Suspicious GT Selector</i> . See Section 6.10.
Suspicious SP Banning Rank	15		
Sufficient TLRus PSP	10	20	Negative effects on <i>Uncolluding MPSPs Detection Rate</i> and <i>Banned Innocent PSPs Rate</i> .
Sufficient TLRus SP	2		
TST Settings. See Sections 5.3 and 5.7 for more details.			
$\tau_{ST-maxLife}$	80		
ST Report Abuse	True		Starting from Milestone 4.2.
PSL ST Selector	false		
Suspicious Nodes ST Selector	false	true	Showed little significance so far
Top <i>TLR_{avg}</i> ST Selector	true	false	Aggressive Selector selecting many <i>Uncolluding M(P)SPs</i> who try to fool Auditor by simple attacks that give them high ranks.
Bottom <i>TLR_{avg}</i> ST Selector	false		
IIR ST Selector	false	true	Showed little significance so far.
TGT Settings. See Sections 5.4 and 5.7 for more details.			
$\tau_{GT-maxLife}$	80		
GT Report Abuse	true		Starting from Milestone 4.2.
PCR Threshold	7	3	Has little importance when $N_{toAbuse_C}$ is low, which is highly expected from optimised Attackers.
PCR Weak Threshold	2	7	Opposite the above note.

Manual Vs. Automatic Defensive Strategies Settings			
Factor	Manual	Automatic	Notes
Ignore Old G Ranks	false	true	
GT Max SP Num	2		
GT Max Size	5	7	Lower <i>GT Max Size</i> values would generate smaller potential Colluding Gs leading to bolder TGT ranks.
PIIL GT Selector	true	false	Expected to boost the Detection Rates with the combination of the other Manual settings.
PSL GT Selector	false	true	Has little effects when <i>PIIL GT Selector</i> is turned on.
Suspicious Nodes GT Selector	false		

TABLE C.1: Manual Vs. Automatic Defensive Strategies Settings

Settings:			ini Norm. High		Norm. GR High		ini Mal. High		Mal. GR High	
Best Defense:		Manual	Settings' Interpretation:			A popular network among Users, (P)SPs, and M(P)SPs.				
Defense	Agent Report?	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
Manual	True	15%	32%	11%	35%	63%	91%	11%	8%	17%
Automatic	True	2%	14%	0%	11%	18%	90%	10%	8%	1%
Settings:			ini Norm. High		Norm. GR Low		ini Mal. High		Mal. GR High	
Best Defense:		Manual	Settings' Interpretation:			A popular network among Users, (P)SPs, and M(P)SPs, but Users and (P)SPs are losing interest in it, due to saturation, better alternatives, and/or the high density of M(P)SPs in the network.				
Defense	Agent Report?	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
Manual	True	10%	19%	12%	28%	84%	82%	14%	8%	7%
Automatic	True	2%	5%	0%	6%	37%	79%	12%	12%	0%
Settings:			ini Norm. Low		Norm. GR High		ini Mal. High		Mal. GR High	
Best Defense:		Manual	Settings' Interpretation:			A bootstrapping network that is gaining rapid popularity among Users and (P)SPs. However, M(P)SPs have already anticipated this success and, hence, they were among the early members to join the network and populate it. Plus, they are still joining at a steady pace.				
Defense	Agent Report?	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
Manual	True	17%	36%	13%	37%	75%	87%	14%	13%	19%
Automatic	True	1%	6%	0%	6%	20%	83%	12%	13%	1%
Settings:			ini Norm. Low		Norm. GR High		ini Mal. Low		Mal. GR High	

Best Defense:		Manual	Settings' Interpretation:			A bootstrapping network that is gaining rapid popularity among Users, (P)SPs and M(P)SPs as well.					
Defense	Agent Report?	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs	
Manual	True	37%	57%	11%	51%	67%	88%	10%	8%	32%	
Automatic	True	8%	22%	0%	19%	25%	84%	10%	10%	3%	
Settings:		ini Norm. Low			Norm. GR Low		ini Mal. Low		Mal. GR High		
Best Defense:		Automatic	Settings' Interpretation:			A bootstrapping network that is gaining rapid popularity among M(P)SPs but not genuine Users and (P)SPs, perhaps due to the network getting polluted with many M(P)SPs.					
Defense	Agent Report?	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs	
Manual	True	42%	52%	12%	47%	85%	68%	13%	13%	39%	
Automatic	False	5%	17%	0%	17%	55%	66%	14%	15%	3%	
Settings:		ini Norm. Low			Norm. GR Low		ini Mal. Low		Mal. GR Low		
Best Defense:		Automatic	Settings' Interpretation:			A bootstrapping network that is not gaining interest from any party, including M(P)SPs. i.e. probably a dying network.					
Defense	Agent Report?	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs	
Manual	True	27%	37%	9%	39%	85%	75%	13%	11%	22%	
Automatic	False	7%	21%	0%	19%	55%	72%	14%	13%	4%	

TABLE C.2: Simulation Stage 4 Detailed Results

In Stage 4, see Subsection 6.6.4, two main Defensive Strategies, *Manual* and *Automatic*, were tested under various environmental settings related to the populations and growth rates of Users, (P)SPs, and M(P)SPs. Table C.1 compares between the setting of the two Strategies while Table C.2 lists the detailed results of their Simulation testing under various Environmental conditions.

Milestone 4.1: Optimising the two groups of Attackers (Uncolluding and Colluding) Against the Manual and Automatic Defensive Strategies

- **Simulated Factors:** *Defense Strategy Flavour Flavour* and the Attackers Factors of Table 6.6.
- **Factors with most Significant Effects:** The *Manual Strategy Flavour Flavour* has huge positive effects on the *Total Agents to Users Rate* and the *Uncolluding MSPs Detection Rate*, over 40% in comparison to those rates generated when *Automatic Strategy Flavour Flavour* is turned on. When it comes to the Attackers effects, they are quite similar to Milestone 3.1.
- **Attack Characteristics:** For the *Uncolluding Attackers*, the toughest simple strategies settings were selected, although the *Uncolluding/Colluding Abuse Bombarding Periods* did not show noticeable effects. For the *Colluding Attackers*, it was interesting to note that the combination of $N_{toAbuse_U} = 5$ and $N_{toAbuse_C} = 1$ would have the same negative effects on the *Colluding MPSPs Detection Rates* but with the advantage of increasing the *Compromised Credentials Rates*. In other words, launching *Weak Colluding* attacks is justified despite the slight risk of abusing a *Testing Agent*. That is specially true when such *Weak Colluding* attacks are combined with high *Users/Credentials Abuse Drop Rates*. Nevertheless, if for any reason the Attackers decided to launch a *Strong Colluding Attack*, it would be better to lower down the values of *Users/Credentials Abuse Drop Rates* in order to increase the *Compromised Credentials Rates*. Interestingly, we have noted that if the Attackers got a way to determine which Defensive Strategy Flavour Flavour is being deployed by the Auditor, see the Reverse Engineering Threat in 5.8.5, and they figured out it was in deed the *Automatic Strategy Flavour Flavour*, shifting from Low Level to High Level *Colluding Attack* settings would generate a very good outcome for the Attackers in terms of increasing the *Compromised Credentials Rates* and decreased *Colluding/Uncolluding MPSPs Detection Rates*.
- **Main Attackers:** In this Milestone we compared 4 Scenarios. First, when the *Manual Strategy Flavour Flavour* is deployed during High Level Attacks, the *Popular Colluding MPSPs* would be the main threat generating around 80% of the total Abuse followed by the *Uncolluding MPSPs* generating around 20% of the total Abuse. Second, when the *Automatic Strategy Flavour Flavour* is deployed during High Level Attacks, the *Popular Colluding MPSPs* would be the main threat generating around 80% of the total Abuse followed by both the *Uncolluding MPSPs* and *Unpopular Colluding MPSPs* where each is generating around 10% of the total Abuse. Third, when the *Manual Strategy Flavour Flavour* is deployed during Low Level Attacks, the *Unpopular Colluding MPSPs* would be the main threat generating around 82% of the total Abuse followed by the *Popular Colluding MPSPs* generating around 13% of the total Abuse. Fourth, when the *Automatic Strategy Flavour Flavour* is deployed during Low Level Attacks, the *Unpopular Colluding MPSPs* would be the main threat generating around 55% of the total Abuse followed by the *Popular Colluding MPSPs* generating around 28% of the total Abuse and the *Uncolluding MSPs* generating around 19% of total Abuse.

- **Defense Characteristics:** We have noted that the *Manual Strategy Flavour Flavour* would increase the *Banned Innocent PSPs Rate* during *Low Level Colluding Attacks*. That is probably because the more trivial the *Colluding Attacks* are, the more Cases they would generate and, hence, more confused the *PIIL GT Selector*, which is utilised by the *Manual Strategy Flavour Flavour*, would get. This confusion would cause it to become a random Selector with a high *Banning Innocent PSPs Rate*. Hence, the *Automatic Strategy Flavour Flavour* is preferred during *Low Level Attacks* due to its modest *Banned Innocent PSPs Rate* and to its low *Active Agents to Users Rate*. In the other hand, during *High Level Attacks*, the Auditor is indifferent to which Strategy Flavour Flavour is being utilised. That is probably because although the *Manual Strategy Flavour Flavour* is doing a better job at increasing the *Uncolluding/Colluding MPSPS Detection Rates*, by utilising the *PIIL GT Selector* and improving the TGT Ranking Weights and settings, it is huge demand of *Testing Agents* makes the *Automatic Strategy Flavour Flavour* an equally appealing alternative. Just like Milestone 3.3, we have noted that the *PIIL GT Selector*, which is utilised by the *Manual Strategy Flavour Flavour*, does not scale; it does an excellent job at detecting *High Level Colluding Attackers* if the network population is small enough. Once a population threshold is crossed, the *PIIL GT Selector* starts to act as a random Detector, see the *Malicious Density Theory 6.8*.
- **Trust Status:** In this Milestone we compared 4 Scenarios. First, when the *Manual Strategy Flavour Flavour* is deployed during *High Level Attacks*, the *Users Trust* reached 93% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 12% and 7% respectively. Second, when the *Automatic Strategy Flavour Flavour* is deployed during *High Level Attacks*, the *Users Trust* reached 80% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 10% and 9% respectively. Third, when the *Manual Strategy Flavour Flavour* is deployed during *Low Level Attacks*, the *Users Trust* reached 65% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 8% and 4% respectively. Fourth, when the *Automatic Strategy Flavour Flavour* is deployed during *Low Level Attacks*, the *Users Trust* reached 70% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 18% and 10% respectively.
- **Auditor Performance:** In this Milestone we compared 4 Scenarios. First, when the *Manual Strategy Flavour Flavour* is deployed during *High Level Attacks*, we got the *Uncoluding MPSPs*, *Uncolluding MSPs*, and *Colluding Unpopular MPSPs* Detection Rates reaching around 11%, 8% and 2% respectively. The *Banned Innocent PSPs Rate* was 0.5% and the *Active Agents to Users Rate* reached about 88%. Second, when the *Automatic Strategy Flavour Flavour* is deployed during *High Level Attacks*, we got the *Uncoluding MPSPs* and *Colluding Popular MPSPs* Detection Rates reaching around 20% and 10% respectively. The *Banned Innocent PSPs Rate* was 1.8% and the *Active Agents to Users Rate* reached about 32%. Third, when the *Manual Strategy Flavour Flavour* is deployed during *Low Level Attacks*, we got the *Uncoluding MPSPs*, *Colluding Popular MPSPs*, *Uncolluding MSPs*, *Colluding Unpopular MPSPs*, and *Colluding Unpopular MSPs* Detection Rates reaching around 100%, 100%, 90%, 70%, and 5% respectively. The *Banned Innocent PSPs Rate* was 16% and the *Active Agents to Users Rate* reached about 86%. Fourth, when the *Automatic Strategy Flavour Flavour* is deployed during *Low Level Attacks*, we got the *Uncoluding MPSPs*, *Colluding Popular MPSPs*, *Uncolluding MSPs*, *Colluding Unpopular MPSPs*, and *Colluding Unpopular MSPs* Detection Rates reaching around 100%, 95%, 40%, and 15%. The *Banned Innocent PSPs Rate* was 5.5% and the *Active Agents to Users Rate* reached about 31%.

- **Final Settings for the following Milestone:** *Uncolluding Abuse Delay Period* = 50, *Uncolluding Abuse Bombarding Period* = 37, *Uncolluding Abuse User/Credntial Drop Rates* = 80, *Colluding Abuse Delay Period* = 50, *Colluding Abuse Delay Period* = 50, *Colluding Abuse User Drop Rate* = 75, *Colluding Abuse Credential Drop Rate* = 25, $N_{toAbuse_U} = 5$, and $N_{toAbuse_C} = 1$.

Milestone 4.2: Optimising the Manual and Automatic Defensive Strategies under variable Populations Scenarios

- **Simulated Factors:** *Defense Strategy Flavour Flavour*, *ST/GT Report Abuse*, *ini Users & (P)SP Nodes*, *Users & (P)SPs Nodes GR*, *ini Uncolluding MPSPs & Colluding M(P)SPs*, *Uncolluding MPSPs & Colluding M(P)SPs GR*, *ini Uncolluding MSPs*, and *Uncolluding MSPs GR*.
- **Factors with most Significant Effects:** The *Manual Defence* has huge positive effects on the *Active Agents to Users Rate* as well as mild positive effects on the *Users Trust*, *M(P)SPs Detection Rates* and the *Banned Innocent PSPs Rate*. The *ini Users & (P)SPs* factor has a significant negative effect on the *Active Agents to Users Rate* as well as a mild positive effect on the *Users Trust*. The *Users & (P)SPs GR* has a significant positive effect on the *Users Trust* as well as significant negative effect on the *Active Agents to Users Rate*.
- **Attack Characteristics:** Fixed since Milestone 4.1.
- **Main Attackers:** In this Milestone, we compared many different populations scenarios. We noticed that in all of the scenarios, the main threat was the *Colluding Popular MPSPs* Attackers with total percentage of Abuse ranging between 60% to 70%. The second threat was coming from the *Uncolluding MPSPs* with total percentage of Abuse ranging from 15% to 30%. Interestingly, we noticed that when both the *ini Users & (P)SPs* and the *ini M(P)SPs* factors are Low, the threat of the *Colluding Unpopular MPSPs* Attackers becomes almost identical to the threat of the *Uncolluding MPSPs* Attackers.
- **Defense Characteristics:** We have noted that both the *ini Users & (P)SPs* and the *Users & (P)SPs GR* have significant positive independent effects on the *Users Trust*. That could be due to the wealth of generated Cases, by a larger population of Users, or due to the increased percentage of innocent (P)SPs, perhaps a simulation bias as explained in Section 6.10. The *Users & (P)SPs GR* also has a milder positive independent effect on the *Active Agents to Users Rate* which could be explained by the fact that the new innocent (P)SPs that are added, at a higher growth rate with higher initial ranks, to the network would be attractive alternatives to the polluted population of (P)SPs that inhabits the network. An alternative explanation might be the wealth of Cases that would be generated by the new Users joining the Network which would, in turn, make it possible to the TGT Detectors to detect the *Colluding MPSPs* without the need to create many GT Agents. The *ini M(P)SP* factor has negative effects on the *Uncolluding/Colluding M(P)SPs Detection Rates* since an increased population of MPSPs would mean many of them would appear, in the reported Cases logs, to be colluding while they are not. Furthermore, the tremendous amount of open Cases would make it unclear who is really guilty among all the MPSPs appearing within those Cases. Surprisingly, the *ini Users & (P)SPs* factor has a negative effect on those *Uncolluding/Colluding M(P)SPs Detection Rates*. That could be probably explained by the fact that an increased percentage of innocent (P)SPs combined with a deployed High *Abuse Delay Period* Attack, the open Cases would contain mostly innocent (P)SPs. Many (P)SPs could appear together in several open Cases making them falsely accused for colluding by the TGT Detectors leading to an increase in the *Banned Innocent PSPs Rate*. However, this negative effect would disappear when

ini M(P)SPs value is High. We have also noted that the *MSP GR* does not have any notable effects on the Monitored Responses. Prior to the simulation, we anticipated that overwhelming the network with MSPs, that are cheap to create by the Attackers, would confuse the Auditor and render its logs useless due to the wealth of Cases generated by those extra MSPs. However, this hypothesis is rejected by this experiment. That might be because the Users are less likely to interact with Unpopular MSPs and the fact that the TGT Detectors do not give much consideration in their analysis to those MSPs. Interestingly, we noted an interaction between Low *ini Users & (P)SPs* and Low *ini M(P)SPs* lead to Low *Users Trust*, High *Uncolluding/Colluding MPSPs Detection Rates*, and High *Banned Innocent PSPs Rate*. This means that the previous Milestones, which were executed in an environment containing that interaction, would be biased toward this setting and may not necessarily represent the equilibrium Regions between the Auditor and the Attackers at different populations settings. This is discussed more in Section 6.10. Table 6.8 summarise the interpretations of each population factor setting along with their observed effects. Table ?? shows the optimal choice of *Defensive Strategies* to be deployed by the Auditor at different populations settings. The observations of Table ?? can be summarised by the following rules:

- When MPSPs Detection is easy, trivial Attackers, use *Automatic Defence*.
- When the *Banning Innocent PSPs Rate* and the *Active Agents to Users Rate* are minimised by either, or all, of High *Users & (P)SPs GR*, High *M(P)SPs GR*, *ini Users & (P)SPs*, or *ini M(P)SPs* factors, use *Manual Defence*.

Another way to look at those observations would be using the following rules:

- At periods of rapid bootstrapping (including M(P)SPs) and/or Positive Saturation, use *Manual Defence*.
 - At periods of slow bootstrapping and/or stable Saturation, use *Automatic Defence*
 - At periods of positive saturation and M(P)SPs losing interest in the Network, use *Automatic Defence*.
- **Trust Status:** Interestingly, we noted that the deployed *Defensive Strategy Flavour Flavour* wont have a noticeable influence on the the Trust observed outputs; it is the populations settings that would matter. The detailed results could be looked at Table C.2.
 - **Auditor Performance:** Similar to the previous Milestones, we noted that the *Manual Strategy Flavour Flavour* is a more aggressive Detector, compared to the *Automatic Strategy Flavour*, that can detect much more M(P)SPs at the cost of extreme rates of *Active Agents to Users* and *Banned Innocent PSPs*. We have also noted that both *Defensive Strategies'* Detection Rates would get boosted at certain population scenarios, when both *ini Users & (P)SPs* and *ini M(P)SPs* factors are set to Low in addition to setting either of *Users & (P)SPs GR* or *M(P)SPs GR* High. Setting both *Users & (P)SPs GR* and *M(P)SPs GR* High would make the *Defensive Strategies* effects even more prevalent. The detailed results could be looked at Table C.2.
 - **Final Settings for the following Milestone:** *Defensive Strategy Flavour* = Manual, *ST/GT report Abuse* = True, *ini Users & (P)SPs* = Low, *Users & (P)SPs GR* = High, *ini M(P)SPs* = Low, *M(P)SPs GR* = High.

C.5 The Milestones of Stage 5: Introducing DGU with Offering Voluntarily DGU Installation Randomly along with a Refinement Iteration

Milestone 5.1: Optimising the two groups of Attackers (Uncolluding and Colluding) against the Manual Defensive Strategy Flavour in a rapid bootstrapping environment after the introduction of DGU

- **Simulated Factors:** *Post DGU Installation rank* and the Attackers Factors of Table 6.6.
- **Factors with most Significant Effects:** The $N_{toAbuse_C}$ has significant positive effects on the *User's Trust* and the *Active Agents to Users Rate*. Both $N_{toAbuse_C}$ and $N_{toAbuse_U}$ have negative effects on the *Colluding Popular MPSPs Detection Rate* as well as mild negative effects on the *Colluding Unpopular MPSPs Detection Rate* either by themselves or by the interaction of their High values. The $N_{toAbuse_U}$ has a mild negative effect on the *Users Trust*. The *Colluding Users/Credentials Abuse Drop Rates* have mild positive effects on the *Users Trust* as well as mild negative effects on the *Colluding Popular MPSPs Detection Rates*. The *Uncolluding Users/Credentials Abuse Drop Rates* have significant negative effects on the *Uncolluding M(P)SPs Detection Rates* as well as mild negative effects on the *Active Agents to Users Rate*. The *Uncolluding Abuse Delay Period* also has a mild negative effect on the *Uncolluding M(P)SPs Detection Rates*.
- **Attack Characteristics:** Generally speaking, the Attacks Characteristics are similar to those observed in Milestone 4.1. One exception is the fact that the *Users/Credentials Drop Rates*, *Abuse Delay Period*, and the $N_{toAbuse_C}$ and $N_{toAbuse_U}$ settings are now having less influence on the *Open Cases Rate* compared to the *Abuse Bombarding Period* influence which becomes dominant. The reason might be the fact that most innocent (P)SPs would eventually choose to *install DGU* voluntarily when asked to do so. Such a behaviour would eliminate those (P)SPs from the Cases and, hence, leave the MPSPs alone in those Cases. That would lead to speeding up the detection process with better accuracy. That is due to the fact that the deployed Defensive Strategy Flavour, *Manual Strategy Flavour* is an aggressive Strategy Flavour that would try to ban even the innocent (P)SPs. Given the fact that innocent PSPs would be rare in the Cases, most of the initially detected (M)PSs would probably be MPSPs, even if they were selected randomly. In case an innocent PSP is mistakenly accused, it would still has the chance to reverse the banning decision by accepting to *install DGU*, or *enabling strict DGU* if it had already *installed DGU*. Still, the *Abuse Bombarding Period* could increase the initial *Banned Innocent PSPs Rate* and, hence, forcing those innocent PSPs to either *install DGU* or getting permanently banned from the network. As a result, MPSPs may get temporarily, or perhaps permanent, relief from getting banned if innocent PSPs kept joining the network at high enough rates where the Cases wont get dominantly populated with MPSPs as described earlier. Another interesting difference from Milestone 4.1 is found in the *Colluding MPSPs* optimal settings. Instead of setting one of the *Users/Credentials Drop Rates* High along with the combination of $N_{toAbuse_U} = 5$ and $N_{toAbuse_C} = 1$, the automatic optimiser for the Attackers opted this time for Low *Users/Credentials Drop Rates* along with increasing $N_{toAbuse_C}$ from 1 to 3. By comparing the initially anticipated outputs of the currently optimised attacks to the earlier settings, they seem to have almost the same effectiveness. However, the main intended advantage of the new setting would be the increase in the *Compromised Credentials Rates*. The reason could be the fact that the more innocent PSPs voluntarily *installing DGU*, the more *Colluding MPSPs* would be exposed in the new Cases. That would mean that the *users/Credentials Drop Rates* wont hide them as effectively as it used to do and, hence, it would be more interesting to increase the *Compromised Credentials rates*. That could be achieved by decreasing the *Users/Credentials Drop Rates* values and

increasing the complexity of the Colluding Attack settings to overcome the lowered effectiveness of the *Users/Credentials Drop Rates* settings. Another possible explanation for the reason why the Attackers decided to reduce their *Users/Credentials Drop Rates* could be the fact that when *Colluding MPSPs* got detected and requested to *install DGU*, they could basically accept the offer and get, in return, a new long life where they can deploy the Advanced Colluding Attack, See Subsection 5.8.7. Hence, there is no serious threat for the *Colluding MPSPs* when they reduce their *Users/Credentials Drop Rates*. Rather, they would just gain increased *Compromised Credentials Rates*. Nevertheless, the actual running of the new settings showed worse performance than previous Milestones. That is, the *MPSPs Detection Rates* got better while the *Compromised Credentials Rates* got worse. That would signal a non-linearity in the *Colluding Attacks Settings* measured effects. In other words, it seems the *Colluding Attackers* should have chosen $N_{toAbuse_C} = 4$ instead of 3 or they should have kept one of the *Users/Credentials Drop Rates* High in order to obtain the desired results. See Section 6.10 for more details.

- **Main Attackers:** The *Popular Colluding MPSPs* is the main threat generating around 72% of the total Abuse followed by the *Uncolluding MPSPs* generating around 15% of the total Abuse.
- **Defense Characteristics:** We have noted that the newly introduced factor, *Post DGU Install Rank* did not show any noticeable effects on the measured outputs. We have also noted that the Defence in this Milestone is worse than what it used to be in Milestone 4.1, before introducing DGU, with the exception of the significantly lowered *Active Agents to Users Rate*. The lowered *Active Agents to Users Rate* might be due to the different environmental settings of this Milestone. Actually, while the introduction of DGU helped in protecting innocent PSPs from being falsely accused of being guilty, a serious shortcome of the *Manual Strategy Flavour* that was chosen in Stage 6.6.4 as the optimal Defensive Strategy Flavour for most of the environmental scenarios, DGU significantly reduced the *Users Trust* and prolonged the network exposure to malicious activities. That is because it gave the accused entities, good and bad ones, second lives by simply accepting to *install DGU*. We initially thought that a possible mitigation would be to lower the *Post DGU Install Rank*, the new life Rank. Hence, we examined the responses with explicitly setting the *Post DGU Install Rank* to be 20%, 50%, and 80%. However, it turned out that this factor does not have any noticeable effects and, hence, it was set to 50% as the average value for the following milestones. Generally speaking, it is good that the *Banned Innocent PSPs Rates* reached 0 at the end of this Milestone. However, it is worrying that most of the Innocent PSPs have already not only *Installed DGU* at the early cycles of the simulation, but also *Enabled Strict DGU*, something that is not expected to be as rapid in real life situations where everyone would be hesitant to deploy such a new technology. This situation is a potential bias source in our next set of simulation Milestones, see Section 6.10 for more details.
- **Trust Status:** The *Users Trust* reached 96% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 9% and 4% respectively.
- **Auditor Performance:** The *Uncolluding MPSPs*, *Uncolluding MSPs*, and *Colluding Popular MPSPs* Detection Rates reached around 45%, 12% and 10% respectively. The *Banned Innocent PSPs Rates* were all 0%. The *Uncolluding MPSPs*, *Uncolluding MSPs*, *Colluding Popular MPSPs*, and *Colluding Unpopular MPSPs* who *Installed DGU* Rates reached around 45%, 12%, 70%, and 40% respectively while the *PSPs* and *SPs* who *Installed DGU* Rates reached around 99% and 98% respectively. The *Uncolluding MPSPs* and the *Colluding Popular MPSPs* who *Enabled Strict DGU* Rates reached around 4%, 6% respectively while the *PSPs* and *SPs* who *Enabled Strict DGU* Rates reached around 95% and 92% respectively. The *Active Agents to Users Rate* reached about 68%.

- **Final Settings for the following Milestone:** *Post DGU Install Rank = 50, Uncolluding Abuse Delay Period = 50, Uncolluding Abuse Bombarding Period = 5, Uncolluding Abuse User/Credential Drop Rates = 80, Colluding Abuse Delay Period = 50, Colluding Abuse Bombarding Period = 50, Colluding Abuse Credential Drop Rate = 20, $N_{toAbuse_U} = 5$, and $N_{toAbuse_C} = 3$.*

Milestone 5.2: Optimising important TST and TGT factors after deploying DGU

- **Simulated Factors:** W_{TGT} , TGT/TST internal Selectors, *PCR threshold, PCR Weak Threshold, and GT Max Size.*
- **Factors with most Significant Effects:** The *PIIL GT Selector* has huge positive effects on the *Users Trust, Active Agents to Users Rate.* It also has significant positive effects on the *Uncolluding MPSPs Detection Rate* and the *Colluding MPSPs Detection Rates.* *Top TLRavg ST Selector* has a huge positive effects on the *Active Agents to Users Rate* as well as a significant positive effect on the *Uncolluding MSPs Detection Rate.* The *Suspicious Nodes ST Selector* also has a significant positive effect on the *Active Agents to Users Rate.* The *PCR Threshold* has mild negative effects on the *Colluding MPSPs Detection Rates.*
- **Attack Characteristics:** Fixed since Milestone 4.1.
- **Main Attackers:** The *Popular Colluding MPSPs* are the main threat generating around 68% of the total Abuse followed by both the *Uncolluding MPSPs* and *Colluding Unpopular MPSPs* each generating around 12% of the total Abuse.
- **Defense Characteristics:** Unlike Milestone 3.3, the *PCR Threshold* here has more noticeable effects than *PCR Weak Threshold.* That is expected given the focus of the current Attackers on Strong Colluding Attacks, setting $N_{toAbuse_C} = 3$. Furthermore, most innocent (P)SPs are now voluntarily *Installing DGU*, meaning that they are automatically excluded from the open Cases logs. Hence, the Auditor would end up with smaller Cases logs to analyse with less Weak Attacks probability. In such a situation, the TGT_{sg} and TGT_{sc} are expected to perform better than TGT_{wc} . In other words, it is advantageous now for the Auditor to deploy more evidence based detection process rather than aggressively, and often randomly, accusing PSPs for open Abuse Cases. Interestingly, we have noted that the *PIIL GT Selector* is the main factor influencing the *Uncolluding MPSPs Detection Rate* rather than the *Top TLRavg ST Selector* as in the previous Milestones. That could be explained by the fact that under the current Defensive Strategy Flavour, the MPSPs are expected to appear in many open Cases in suspicious patterns since the logs are now smaller, because most innocent PSPs have already *Installed DGU.* Hence, the *Uncolluding MPSPs* would get accused for potential colluding although they are not really colluding. At the end, the automatic optimiser preferred turning off the *Top TLRavg ST Selector* to decrease the *Active Agents to Users Rate* at the cost of reducing the *Uncolluding MSPs Detection Rate* to almost 0, which are not a major threat anyways.
- **Trust Status:** The *Users Trust* reached 96% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 7% and 3% respectively.
- **Auditor Performance:** The *Uncolluding MPSPs, Popular Colluding MPSPs, and Unpopular Colluding MPSPs* Detection Rates reached around 62%, 42%, and 30% respectively. The *Banned Innocent PSPs Rates* reached about 0% and the *Active Agents to Users Rate* reached about 50%.

- **Final Settings for the following Milestone:** $W_{TGT} = 80$, all ST Selector turned off except for *IIR ST Selector*, all GT Selectors are turned on except for *Suspicious GT Selector*, $PCR Threshold = 2$, $PCR Weak Threshold = 7$, and $GT Max Size = 7$. It should be noted that due to human error which was explored toward the end of the simulation project, W_{TLRavg} was changed from 4 to 8 while W_{TST} was changed from 16 to 12 by the end of this Milestone. This change is a mistake that should not have happened. Still, we do not think it would have noticeable effects on the integrity of this simulation project, see Section 6.10 for more details.

Milestone 5.3: Optimising the two groups of Attackers (Uncolluding and Colluding) against the optimised Auditor deploying DGU

- **Simulated Factors:** *Post DGU Installation rank* and the Attackers Factors of Table 6.6.
- **Factors with most Significant Effects:** Similar to Milestone 5.1 but with generally milder effects. An exception is the *Uncolluding Users/Credentials Abuse Drop Rates* which now have significant negative effects on the *Uncolluding M(P)SPs Detection Rates* as well as mild negative effects on the *Colluding MPSPs Detection Rates*. That is expected since turning off the *Top TLRavg ST Selector* had the side effect of reducing the *Uncolluding M(P)SPs Detection Rates* leading to an increase volume of open Cases logs. This increase would negatively affect the efficiency of the accurate TGT_{sg} and TGT_{sc} Detectors that were relied upon in Milestone 5.2. This scenario means that High *Users/Credentials Drop Rates* would cause even more confusion and trouble to the currently deployed accurate Detectors.
- **Attack Characteristics:** Generally speaking, the Attacks Characteristics are similar to those observed in Milestone 4.1. Even the changes made in Milestone 5.1 where the *Colluding Users/Credentials Drop Rates* were both minimised and the increasing of factor $N_{toAbuseC}$ from 1 to 3 are now undone.
- **Main Attackers:** The *Popular Colluding MPSPs* is the main threat generating around 75% of the total Abuse followed by the *Uncolluding MPSPs* generating around 18% of the total Abuse.
- **Defense Characteristics:** Similar to Milestone 5.1, we have noted that the factor *Post DGU Install Rank* did not show any noticeable effects on the measured outputs. We have also noted that the Defence in this Milestone is better than Milestone 5.1, except for the *Uncolluding MSPs Detection Rate*. Still, the defense in this Milestone is worse than Milestone 4.1, except for the *Active Agents to Users* and the *Uncolluding MPSPs Detection Rate*. It seems that deploying *PIIL GT Selector* along with lower $PCR Threshold$ under an environment where most innocent (P)SPs voluntarily *install DGU* helped a lot in reducing the *Active Agents to Users Rate* and *Uncolluding MPSPs Detection Rate* without increasing the *Banned Innocent PSPs Rates*. Still, giving the *Colluding MPSPs* a fresh life deeply affected the detection rates of colluding. To be accurate, it is not the *Post DGU Install Rank* aspect of the new life that aided the *Colluding MPSPs* to survive, according to our analysis in both this Milestone and in Milestone 5.1. Rather, it is the advanced colluding attack that we, unwillingly, push the *Colluding MPSPs* to adopt after we force them to *Install DGU*, see Subsection 5.8.7 for more details.
- **Trust Status:** The *Users Trust* reached 85% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 8% and 9% respectively.
- **Auditor Performance:** The *Uncolluding MPSPs Detection Rate* reached around 20% while the rest of *M(P)SPs Detection Rates* were almost 0%. The *Banned Innocent PSPs Rates* were all 0%. The *Uncolluding MPSPs*, *Uncolluding MSPs*, *Colluding Popular MPSPs*, and *Colluding Unpopular MPSPs* who *Installed DGU Rates* reached around 21%, 2%, 17%, and 5% respectively while the *PSPs* and *SPs*

who *Installed DGU Rates* both reached around 98%. The *M(P)SPs* who *Enabled Strict DGU Rates* were all below 1% while the *PSPs* and *SPs* who *Enabled Strict DGU Rates* both reached around 91%. The *Active Agents to Users Rate* reached about 50%.

- **Final Settings for the following Milestone:** *Post DGU Install Rank* = 50, *Uncolluding Abuse Delay Period* = 50, *Uncolluding Abuse Bombarding Period* = 50, *Uncolluding Abuse User/Credntial Drop Rates* = 80, *Colluding Abuse Delay Period* = 50, *Colluding Abuse Bombarding Period* = 5, *Colluding Abuse User Drop Rate* = 24, *Colluding Abuse Credential Drop Rate* = 80, $N_{toAbuse_U}$ = 5, and $N_{toAbuse_C}$ = 1.

C.6 The Milestones of Stage 6: Removing the Voluntarily DGU Installation Option along with a Refinement Iteration

Milestone 6.1: Optimising the two groups of Attackers (Uncolluding and Colluding) against the optimised Auditor deploying DGU without offering voluntary DGU installation

- **Simulated Factors:** *Post DGU Installation rank* and the Attackers Factors of Table 6.6.
- **Factors with most Significant Effects:** Similar to Milestone 5.3.
- **Attack Characteristics:** Generally speaking, the Attacks Characteristics are similar to those observed in Milestone 5.3 but with slightly increased *Colluding Users/Credentials Abuse Drop Rates*. The reason for this slight increase might be the fact that the automatic optimiser choose to reduce the *Post DGU Install Rank* from 50% to 20%. The new lower value would mean that the Auditor would get more aggressive at banning suspicious nodes and, hence, the *Colluding MPSPS* should get extra careful by reducing their Abuse rates.
- **Main Attackers:** The *Popular Colluding MPSPs* is the main threat generating around 68% of the total Abuse followed by the *Uncolluding MPSPs* generating around 20% of the total Abuse.
- **Defense Characteristics:** Similar to Milestone 5.3 but with a slight surge in the *Banned Innocent PSPs Rate "Refused to Install DGU"*. The reason for the surge in the *Banned Innocent PSPs Rate "Refused to Install DGU"* could be the fact that, unlike Stage 6.6.5 where most PSPs would voluntarily *install DGU* prior to appearing in open Cases logs, PSPs now appear in the open Cases logs just like the situation before we introduce DGU in Stage 6.6.5. For that, PSPs are now falsely accused to be acting maliciously and, hence, are forced to *Install DGU* or getting banned.
- **Trust Status:** The *Users Trust* reached 94% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 10% and 7% respectively.
- **Auditor Performance:** The *Uncoluding MPSPs Detection Rate* reached around 21% while the rest of *M(P)SPs Detection Rates* were almost 0%. The *Banned Innocent PSPs Rate "Refused to Install DGU"* reached around 4%. The *Uncolluding MPSPs*, *Uncolluding MSPs*, *Colluding Popular MPSPs*, and *Colluding Unpopular MPSPs* who *Installed DGU Rates* reached around 21%, 0%, 23%, and 14% respectively while the *PSPs* who *Installed DGU Rate* reached around 7%. The *M(P)SPs* and *Colluding*

Popular MPSPs who Enabled Strict DGU Rates were both around 1% while the PSPs and SPs who Enabled Strict DGU Rates both were 0%. The Active Agents to Users Rate reached about 53%.

- **Final Settings for the following Milestone:** Post DGU Install Rank = 20, Uncolluding Abuse Delay Period = 50, Uncolluding Abuse Bombarding Period = 49, Uncolluding Abuse User/Credntial Drop Rates = 80, Colluding Abuse Delay Period = 50, Colluding Abuse Bombarding Period = 5, Colluding Abuse User Drop Rate = 56, Colluding Abuse Credential Drop Rate = 74, $N_{toAbuse_U} = 5$, and $N_{toAbuse_C} = 1$.

Milestone 6.2: Optimising important TST and TGT factors after deploying DGU without offering voluntary DGU installation

- **Simulated Factors:** W_{TGT} , W_{sc} , W_{wc} , TGT/TST internal Selectors, PCR threshold, PCR Weak Threshold, and GT Max Size, .
- **Factors with most Significant Effects:** The W_{TGT} is the dominant influencing factor with a mild negative effect on the Users Trust as well as significant positive effect on the Uncolluding MPSPs Detection Rate, the Colluding Popular MPSPs Rate, and the Banned Innocent PSPs “Refused to Install DGU” Rate. The interaction of High W_{TGT} and turned off PIIL GT Selector as well as the interaction of High W_{TGT} and Low PCR Weak Threshold both have positive effects on the MPSPs Detection Rates and the Banned Innocent PSPs Rates. The PCR Weak Threshold have mild negative effects on the Users Trust, the Uncolluding MPSPs Detection Rate, the Colluding MPSPs Detection Rates and the Banned Innocent PSPs. The interaction of Low PCR Weak Threshold and turned off PIIL GT Selector has a mild positive effects on the Colluding Popular MPSPs Detection Rates as well as mild negative effects on the Users’ Trust. The Top TLRAvg ST Selector has huge positive effects on the Active Agents to Users Rate and a significant positive effect on the Uncolluding MSPs Detection Rate. The interaction of setting one of Suspicious ST Selector or Top TLRAvg ST Selector High while setting the second Low has mild positive effect on the Users Trust and the Colluding Popular MPSPs Detection Rates.
- **Attack Characteristics:** Fixed since Milestone 6.1.
- **Main Attackers:** The Popular Colluding MPSPs are the main threat generating around 75% of the total Abuse followed by both the Uncolluding MPSPs and Colluding Unpopular MPSPs each generating around 9% of the total Abuse.
- **Defense Characteristics:** Unlike Milestone 5.2, in this Milestone where no (P)SPs are willing to voluntarily Install DGU, the PIIL records would get so large that the PIIL GT Selector along with its associated Low PCR Threshold would be of little value. Instead, the PCR Weak Threshold would be a more powerful Detector. Combined with High W_{TGT} value, this Detector would get extra authority to ban PSPs even more faster. This new setting causes reduced Users Trust. Nevertheless, this reduction appears to be temporarily since the curve of Compromised Credentials goes steeply high before starting to go sharply down toward the end of the simulation. If we run the simulation for an extended period, we anticipate the Users Trust level to get equivalent or even better than Milestone 5.2. However, the Banned Innocents PSPs Rates are so bad because the PCR Weak Threshold is a Weak Detector that would accuse many innocents that did not voluntarily Install DGU. Another interesting observation is the fact that the interaction between Suspicious ST Selector and Top TLRAvg ST Selector would have an optimised value when only one of them is switched on. The reason for this odd interaction, which would boost the MPSPs Detection Rates, is the fact that under the other optimised settings,

the main factor determining who would be ranked suspicious, and then gets eligible for selection by the *Suspicious ST Selector*, would be the *PCR Weak Threshold*. Hence, this Selector would eliminate many *Uncolluding MPSPs* quickly from the Cases logs making the task of the rest of Selectors easier as they would be dealing with smaller sizes logs. The same effect could be achieved using the *Top TLRavg ST Selector*. Hence, turning on both of those Selectors would be a slightly bad idea because that would remove most of the *Uncolluding MPSPs* from the logs leaving smaller set of logs. Less logs to evaluate makes the *PCR Weak Threshold* unable to make definite guesses about who are the MPSPs. In other words, the *PCR Weak Threshold* is a good Detector only when dealing with large and hard to analyse logs. At the end, it is kind of a Random Detector that works better in heavily polluted networks with high MPSPs Density, see Section 6.8. In fact, that would explain why turning on the *PIIL GT Selector* would negatively affect the Detection Rates in this milestone. That is because the *PIIL GT Selector* would detect some MPSPs but, at the same time, reduce the number of open Cases logs making the task of the *PCR Weak Threshold* even harder. Although the *MPSPs Detection Rates* are stunning in this Milestone, the *Banned Innocent PSPs Rates* are so high and unacceptable. To reduce the *Banned Innocent PSPs Rates* without reducing the *MPSPs Detection Rates*, the Auditor should either convince more (P)SPs to voluntarily *Install DGU* or try to compromise some of the optimised settings. We tried the latter solution by manually adjusting the internal weights of the automatic optimiser so that it doubles the importance of reducing the *Banned Innocent PSPs Rates*, see Subsection 6.5.3. The automatic optimiser then suggested reducing the W_{TGT} to 51% instead of 80%. However, we realised that despite the achieved reduction in the *Banned Innocent PSPs Rates*, the Auditor failed to sustain the good *MPSPs Detection Rates* against the prediction made by the automatic optimiser. That reveals non-linearity in the measured effects of the W_{TGT} , see Section 6.10. For that, we run the simulation with fixed values except for the W_{TGT} which we gradually increased it from 51% to 80% by steps equivalent to half the space between the current run and the final value. It turns out that the critical point where the *Uncolluding MPSPs Detection Rate* would get boosted is when we set W_{TGT} around 76%. However, no good *Colluding MPSPs Detection Rates* improvement happens until W_{TGT} reaches its maximum value of 80%.

- **Trust Status:** The *Users Trust* reached 94% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* both reached around 6%.
- **Auditor Performance:** The *Uncolluding MPSPs*, *Popular Colluding MPSPs*, and *Unpopular Colluding MPSPs* Detection Rates reached around 71%, 30%, and 20% respectively. The *Banned Innocent PSPs* “*Refused to Install DGU*” and the *Banned Innocent PSPs* “*Refused to Enable Strict DGU*” reached about 24% and 6% respectively. The *Uncolluding MPSPs*, *Colluding Popular MPSPs*, and *Colluding Unpopular MPSPs* who *Installed DGU* Rates reached around 71%, 72%, and 73% respectively while the *PSPs* who *Installed DGU* Rate reached around 34%. The *Colluding Popular MPSPs* and *Colluding Unpopular MPSPs* who *Enabled Strict DGU* Rates were around 30% and 21% respectively while the *PSPs* who *Enabled Strict DGU* Rate was about 3%. The *Active Agents to Users Rate* reached about 31%.
- **Final Settings for the *Extreme DGU Strategy Flavour*:** $W_{TGT} = 80$, $W_{sc} = 30$, $W_{wc} = 50$, all ST Selector turned off except for *Suspicious ST Selector*, all GT Selectors are turned on except for *PSL GT Selector*, *PCR Threshold* = 2, *PCR Weak Threshold* = 2, and *GT Max Size* = 2.

Milestone 6.3: Optimising important TGT factors after deploying DGU without offering voluntary DGU installation in a slowly bootstrapping network among Normal nodes but highly popular among malicious nodes

- **Simulated Factors:** W_{TGT} , W_{sc} , W_{wc} , *PIIL GT Selector*, *PCR threshold*, *PCR Weak Threshold*, and *GT Max Size*, .
- **Factors with most Significant Effects:** The *PIIL GT Selector* has a huge positive effect on the *Active Agents to Users Rate* as well as a significant positive effect on the *Users Trust*. The *PCR Weak Threshold* has mild negative effects on the *Users Trust* the *Uncolluding MPSPs Detection Rate*, and the *Banned Innocent PSPs “Refused to Install DGU” Rate*. The interaction of High *PCR Weak Threshold* and the turned off *PIIL GT Selector* has a huge negative effect on the *Active Agents to Users Rate* as well as mild negative effect on the *Users’ Trust*. Both the W_{TGT} by itself and the interaction of High W_{TGT} and Low *PCR Weak Threshold* have a mild positive effect on the *Uncolluding MPSPs Detection Rate* as well as a mild negative effect on the *Banned Innocent PSPs “Refused to Install DGU” Rate*. The interaction of Low W_{TGT} and turned off *PIIL GT Selector* has a mild negative effect on the *Uncolluding MPSPs Detection Rate* and the *Banned Innocent PSPs “Refused to Install DGU” Rate*. The *PCR Threshold* has a mild negative effect on the *Uncolluding MPSPs Detection Rate*. The Interaction of turned on *PIIL GT Selector* and Low *PCR Threshold* has a mild positive effect on the *Uncolluding MPSPs Detection Rate*.
- **Attack Characteristics:** Fixed since Milestone 6.1.
- **Main Attackers:** The *Popular Colluding MPSPs* are the main threat generating around 60% of the total Abuse followed by the *Uncolluding MPSPs* generating around 27% of the total Abuse.
- **Defense Characteristics:** In comparison to Milestone 6.2, the responses in the new environment of this Milestone are generally less sensitive to variations in the defensive factors. However, the combination of turned on *PIIL GT Selector* with Low *PCR Threshold*, which we could call the Strong Colluding Detector, is now more effective at detecting MPSPs and, hence, it effects exceeded those of the Weak Colluding Detector: Low *PCR Weak Threshold* combined with High W_{TGT} . Actually, the Strong Colluding Detector detection effects are now reversed from Milestone 6.2. That is, those effects are now positive rather than negative. That would indicate that the Strong Colluding Detector has switched from its *Low Density Detection Region* to its *Threshold Detection Region*, see Section 6.8. Surprisingly, the automatic optimiser did not turn on the *PIIL GT Selector* despite its positive effects on the *MPSPs Detection Rates*. It looks as if those positive effects would not justify the high cost of increased *Active Agents to Users Rate*. Rather, the optimiser prefers to continue relying on the Weak Colluding Detector combined with the *PSL GT Selector* that would basically put all the MPSPs under a Weak Collective test. Given the large population of MPSPs, this Selector turns out to be very effective in detecting trivial MPSPs, not those deploying Advanced Colluding Attacks. We have also observed that there is a non-linearity in the observed effects of the *PCR Weak Threshold*. Actually, through running simulation with increasing steps of that factor values, we found that there is a critical *PCR Weak Threshold* point, around 4.5, where the Weak Colluding Detector would render useless. In such a case, the Strong Colluding Detector would be the main MPSPs Detector with a better accuracy but reduced efficiency, increased *Active Agents to Users Rate*, as well as effectiveness, reduced Detection Rates.
- **Trust Status:** The *Users Trust* reached 86% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 13% and 10% respectively.
- **Auditor Performance:** The *Uncolluding MPSPs Detection Rate* reached around 62% while the rest of the Detection rates were almost 0%. The *Banned Innocent PSPs “Refused to Install DGU”* reached about 23%. The *Uncolluding MPSPs*, *Colluding Popular MPSPs*, and *Colluding Unpopular MPSPs* who

Installed DGU Rates reached around 59%, 60%, and 52% respectively while the PSPs who Installed DGU Rate reached around 30%. The (M)PSPs who Enabled Strict DGU Rates were all 0%. The Active Agents to Users Rate reached about 69%.

Milestone 6.4: Optimising the two groups of Attackers (Uncolluding and Colluding) against the optimised Auditor of this Stage

- **Simulated Factors:** Post DGU Installation rank and the Attackers Factors of Table 6.6.
- **Factors with most Significant Effects:** The $N_{toAbuse_C}$ has a significant positive effect on the Users Trust and mild positive effects on the Active Agents to Users Rate while it has a significant negative effect on the Colluding MPSPs Detection Rates and a mild negative effect on Banned Innocent PSPs “Refused to Enable Strict DGU” Rate. The Uncolluding Users/Credentials Abuse Drop Rates have a mild positive effects on the Users Trust and mild negative effects on the Uncolluding M(P)SPs Detection Rates as well as mild negative effects on the Colluding MPSPs Detection Rates. Nevertheless, the Uncolluding Users/Credentials Abuse Drop Rates now have less influence than they used to have in Milestone 6.1. The Colluding Users/Credentials Abuse Drop Rates have a mild positive effect on the Users Trust Rate. The Uncolluding Abuse Delay Period has mild negative effects on the Active Agents to Users Rate and the Uncolluding MSPs Detection Rate.
- **Attack Characteristics:** In this Milestone, it seems that the aggressive Extreme DGU Strategy Flavour is forcing the automatic optimiser to raise the complexity of the colluding attack by raising the value of $N_{toAbuse_C}$ from 1 to 5. That is essential to maintain Low MPSPs Detection Rates. It should be noted however that in Milestone 5.1 we figured out the presence of non-linearity in the measured effects of the $N_{toAbuse_C}$ factor and, hence, a lower value of it may do the trick for the Attackers’ automatic optimiser. Since the Extreme DGU Strategy Flavour of Milestone 6.2 does not rely on the PIIL GT Selector but, instead, on PCR Weak Threshold, the Users/Credentials Abuse Drop Rates are now of less influence in the measured effects since their main goal is to trick the PIIL GT Selector which is not utilised now. That would explain the decision of the automatic optimiser to raise the complexity of the colluding attacks while minimising the Colluding Users/Credentials Abuse Drop Rates. That would also explain why the Colluding Popular MPSPs are not the main threat now to the network since they are more reluctant to send SPam due to their complex colluding settings.
- **Main Attackers:** The Uncolluding MPSPs is the main threat generating around 48% of the total Abuse followed by the Colluding Popular MPSPs generating around 30% of the total Abuse.
- **Defense Characteristics:** We have noted in this Milestone that the Active Agents to Users Rate is considerably lower than it used to be in Milestone 6.1 due to turning off many Agents Selectors. However, this output is now more sensitive to variations in the Uncolluding Abuse Delay period since the Suspicious ST Selector is now active. That is because the Suspicious ST Selector is triggered by new (P)SPs being classified as suspicious through the TLRavg Approach, which is the main Defence Strategy Flavour that the Abuse Delay Period tries to evade. By having large Delay values, it would take longer to receive Cases and to lower down the suspicious (P)SPs ranks. Further, the larger logs generated by the Delay attacks means that the ranks lowering process would get even slower and, hence, less the Suspicious ST Selector would create less ST Agents. The Colluding Abuse Delay Period is not as sensitive because complex Colluding Attacks would naturally be delayed until all the colluding condition are fulfilled, regardless of the preset Colluding Abuse Delay Period.

- **Trust Status:** The *Users Trust* reached 98% while the compromised rates of *Share with Top Credentials* and *Share with Any Credentials* reached around 5% and 1% respectively.
- **Auditor Performance:** The *Uncolluding MPSPs*, *Uncolluding MSPs*, *Colluding Popular MPSPs*, and *Colluding Unpopular MPSPs* Detection rates reached around 59%, 2%, 4%, and 4% respectively. The *Banned Innocent PSPs Rate “Refused to Install DGU”* reached around 16% while the *Banned Innocent PSPs Rate “Refused to Enable Strict DGU”* reached around 2%. The *Uncolluding MPSPs*, *Colluding Popular MPSPs*, and *Colluding Unpopular MPSPs* who *Installed DGU* Rates reached around 60% while the *Uncolluding MSPs who Installed DGU* reached around 2% while the *PSPs who Installed DGU* Rate reached around 27%. The *M(P)SPs* and *Colluding Popular MPSPs* who *Enabled Strict DGU* Rates reached around 6% and 4% respectively while the *PSPs* and *SPs* who *Enabled Strict DGU* Rates both were almost 0%. The *Active Agents to Users Rate* reached about 34%.
- **Final Settings for the Extreme DGU associated Attack:** *Post DGU Install Rank* = 20, *Uncolluding Abuse Delay Period* = 5, *Uncolluding Abuse Bombarding Period* = 50, *Uncolluding Abuse User/Credential Drop Rates* = 80, *Colluding Abuse Delay Period* = 37, *Colluding Abuse Bombarding Period* = 50, *Colluding Abuse User/Credential Drop Rates* = 20, $N_{toAbuse_U}$ = 5, and $N_{toAbuse_C}$ = 5.

C.7 The Milestones of Stage 7: Comparing the Performance of vDGU and DGU

Settings:		ini Norm. High		Norm. GR High		ini Mal. High		Mal. GR High	
Best Defense:	None	Settings' Interpretation:		A popular network among Users, (P)SPs, and M(P)SPs.					
Defense	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
vDGU	0%	0%	2%	20%	28%	88%	7%	7%	0%
DGU	0%	0%	1%	14%	36%	92%	8%	11%	1%
Settings:		ini Norm. High		Norm. GR Low		ini Mal. High		Mal. GR High	
Best Defense:	None	Settings' Interpretation:		A popular network among Users, (P)SPs, and M(P)SPs, but Users and (P)SPs are losing interest in it, due to saturation, better alternatives, and/or the high density of M(P)SPs in the network.					
Defense	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
vDGU	0%	0%	2%	9%	50%	80%	10%	10%	1%
DGU	0%	0%	0%	11%	64%	89%	11%	12%	1%
Settings:		ini Norm. Low		Norm. GR High		ini Mal. High		Mal. GR High	
Best Defense:	None	Settings' Interpretation:		A bootstrapping network that is gaining rapid popularity among Users and (P)SPs. However, M(P)SPs have already anticipated this success and, hence, they were among the early members to join the network and populate it. Plus, they are still joining at a steady pace.					
Defense	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
vDGU	1%	3%	1%	27%	53%	87%	10%	10%	0%
DGU	1%	3%	1%	37%	54%	92%	10%	12%	8%
Settings:		ini Norm. Low		Norm. GR High		ini Mal. Low		Mal. GR High	

Best Defense:	None	Settings' Interpretation:		A bootstrapping network that is gaining rapid popularity among Users, (P)SPs and M(P)SPs as well.					
Defense	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
vDGU	1%	3%	1%	24%	50%	87%	9%	9%	0%
DGU	1%	3%	0%	35%	56%	91%	9%	11%	8%
Settings:		ini Norm. Low		Norm. GR Low		ini Mal. Low		Mal. GR High	
Best Defense:	None	Settings' Interpretation:		A bootstrapping network that is gaining rapid popularity among M(P)SPs but not genuine Users and (P)SPs, perhaps due to the network getting polluted with many M(P)SPs.					
Defense	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
vDGU	1%	2%	2%	14%	72%	75%	12%	11%	0%
DGU	1%	4%	1%	30%	78%	83%	12%	13%	8%
Settings:		ini Norm. Low		Norm. GR Low		ini Mal. Low		Mal. GR Low	
Best Defense:	None	Settings' Interpretation:		A bootstrapping network that is not gaining interest from any party, including M(P)SPs. i.e. probably a dying network.					
Defense	UnPop. Detct.	Pop. Detct.	MSPs Detct.	MPSPs Detct.	Agents / Users	Trust	Cmp. shrTop Crd.	Cmp. shrAny Crd.	Banned Inoc. PSPs
vDGU	1%	2%	2%	24%	75%	80%	12%	11%	0%
DGU	1%	3%	2%	30%	80%	87%	12%	13%	6%

TABLE C.3: Simulation Stage 7 Detailed Results

In Stage 7, see Subsection 6.6.7, two main Defensive Strategies, *vDGU* and *DGU*, were tested under various environmental settings related to the populations and growth rates of Users, (P)SPs, and M(P)SPs. Table C.3 lists the detailed results of their Simulation testing under various Environmental conditions.

Milestone 7.1: Optimising the vDGU and DGU Defensive Strategies under variable Populations Scenarios

- **Simulated Factors:** *Defense Strategy Flavour*, *ini Users & (P)SP Nodes*, *Users & (P)SPs Nodes GR*, *ini M(P)SPs*, and *M(P)SPs GR*.
- **Factors with most Significant Effects:** The *Users & (P)SPs Nodes GR* factor has a negative significant effect on the *Active Agents to Users Rate* as well as a mild positive effect on the *Users Trust*. The *ini Users & (P)SPs* factor has a mild negative effect on the *Active Agents to Users Rate*.
- **Attack Characteristics:** Fixed since Milestone 5.3.
- **Main Attackers:** In this Milestone, we compared many different populations scenarios. We noticed that in the scenarios when the *ini M(P)SPs* is High, the main threat was the *Colluding Popular MPSPs* Attackers with total percentage of Abuse around 62% while the second threat was coming from the *Uncolluding MPSPs* with total percentage of Abuse ranging around 28%. When the *ini M(P)SPs* is Low, we noticed that the threat of the *Colluding Popular MPSPs* Attackers raised to becomes around 72% while the threat of the *Uncolluding MPSPs* decrease to get around 15%.
- **Defense Characteristics:** We have noted that the measured Defensive Strategies' effects are milder than what we have seen in Milestone 4.2, i.e. before introducing *DGU*. That could be due to the stability and extra moderation in the Detection process, by offering to *Install DGU* before banning a suspicious (P)SP. That would mean that feeding the Auditor with extra logs, by increasing the populations, is not enough to improve the *MPSPs Detection Rates* significantly since *DGU* is giving new lives for guilty M(P)SPs who agrees to *Install DGU*. Once *Colluding MPSPs* agrees to *Install DGU*, they would continue their malicious activities using more Advanced Colluding Settings as described in 5.8.7. We have also noted that that both the *Users & (P)SPs GR* and *ini Users & (P)SPs* factors have positive independent effects on the *Active Agents to Users Rate*. That could be explained by the fact that the new innocent (P)SPs that are joining the network with high *ini Ranks* would be attractive alternatives to the polluted population of M(P)SPs that inhabits the networks. An alternative explanation might be the wealth of Cases that would be generated by the new Users. Particularly speaking, the new PIIL records, added by the new Users, would reduce the need for creating new Testing Agents to generate new logs. We have also noted that the *ini Users & (P)SPs* and *Users & (P)SPs GR* factors would have mild effects reducing the *Compromised Share with Any Credentials* while the *ini M(P)SPs* factor would have a milder opposite effect. The reason for the *ini Users & (P)SPs* and *Users & (P)SPs GR* reduction effects could be the fact that the more genuine (P)SPs are available in the network, the less likely Users would interact with the M(P)SPs. Moreover, the less Users interacting with M(P)SPs, the longer it would take them to interact with the minimum MPSPs required to fulfil the Colluding condition.
- **Trust Status:** Generally speaking, the *vDGU Strategy Flavour* would have worse *Users Trust* values than the *DGU Strategy Flavour* by 4% to 9%. The detailed results could be looked at Table C.3.

- **Auditor Performance:** Since the *Top TLRavg ST Selector* is turned off, the *Uncolluding MSPs Detection Rate* is almost 0% by both Strategies. Further, since the *Colluding MPSPs* deploy Advance Colluding Settings after they *Install DGU*, the *Colluding MPSPs Detection Rates* are also close to 0% by both Strategies. The *Uncolluding MPSPs Detection Rate* is generally better by the *DGU Strategy*. Regardless of the Strategy Flavour, that Rate gets boosted when the *ini Users& (P)SPs* is Low, something that could be explained by mapping the Density zones described in Section 6.8. When it comes to the *Active Agents to Users*, the *vDGU Strategy Flavour* demand less Agents by percentages ranging from 1% to 14%. I highly saturated network the *Active Agents to Users Rate* would be around 32% while that rate would be around 57% when the genuine Users and (P)SPs start to lose interest in the saturated networks. In bootstrapping networks, that Rate would raise to around 75%. When it comes to the *Banned Innocent PSPs Rates*, the *vDGU Strategy Flavour* would generate 0% while the *DGU Strategy* would generate around 8%. The detailed results could be looked at Table C.3.

C.8 The Milestones of Stage 8: Evaluating the Different Defensive Strategies by ANOVA-Testing

An important note about results presented in the below Milestones is their division into three main categories:

- **Important Observations:** These are some important observations which are helpful to understand how we come to our conclusions when we compared the several Defensive Strategies in Section 6.9.
- **General Notes:** These are some notes that are applicable to certain Strategies regardless of the Density Point where it is being evaluated at.
- **Malicious Density Theory Notes:** These notes are some specific notes to give us hints on how the different Strategies are behaving at different Density Points. These are crucial to confirm some of our initial hypothesis presented in the Malicious Density Theory of Section 6.8 or to show some non-conformance with the Theory which would suggest more improvements to it.

Milestone 8.1: ANOVA testing the Strategies effects on the *Users' Trust*

- **Important Observations:**
 - The fact that the *GPD Strategy Flavour* is performing poorly, worse than the *None Strategy Flavour* in many cases, confirms our observation that this Strategy Flavour is useless and should not be adopted by a wise Auditor unless it is anticipated that the Attackers population is made

up by a majority of amateur Attackers who do not even know how to utilise the Simple Attacking Strategies of 5.8.6.

- The little effects of the *Users' Ignorance Rate* means that the evaluated Strategies are tolerant to User's ignorance and can perform to their full potential even with a fraction of the real Abuse Cases reported. Maybe that is a result of the Attackers' adoption of complex Attacking settings where the reported Cases, even when the *Users' Ignorance Rate* is Low, would be so large making it indifferent to the large Cases that would be received from log an ignorant User. Nevertheless, the fact that the *GPD Strategy Flavour* would perform worse than the *None Strategy Flavour* when the *Users' Ignorance Rate* is High confirms this fact since this Strategy Flavour assumes small Cases in order to identify suspicious (P)SPs to accuse based on the chronological order of their appearing in those small Case, see Section 5.2.

- **Malicious Density Theory Notes:**

- As expected, the Lower the Malicious Density is, the better the *None Strategy Flavour* would perform, since there would not be major Trust Threats.
- The *Automatic Strategy Flavour* is doing 10% worse in the *S1 Density* compared to the *S2 Density*.
- In both the *S4 Density* and the *S3 Density*, i.e. the Low Density Region, the *ST*, *Manual*, and *DGU Strategies* are doing 10% better than in the High Density Region while the *Automatic Strategy Flavour* is better by 30% and the *vDGU Strategy Flavour* is better by 15%. These good figures might be due to the fact that the less the Malicious Density is, the less Normal Users would deal with the Malicious Nodes who would compromise their credentials causing their Trust to drop.
- In all environmental scenarios, the *Extreme DGU Strategy Flavour* is doing significantly better than the rest of the Strategies. That is specially true in the *S1 Density* and *S2 Density*, i.e. near the *Threshold Region*. That would be at the expense of High *Banned Innocent PSPs Rates*.

Milestone 8.2: ANOVA testing the Strategies effects on the *Active Agents to Users Rate*

- **Important Observations:**

- Similar to Milestone 8.1, the Defensive Strategies were resilient to varying *Users' Ignorance Rate* which is good.

- **General Notes:**

- Both the *None* and *GPD Strategies* do not generate Agents. Nevertheless, both are almost useless and, hence, this is not really an incentive to deploy either Strategy Flavour.

- **Malicious Density Theory Notes:**

- The significantly least Strategy Flavour in creating Agents is the *Automatic Strategy Flavour* followed by the *Extreme DGU Strategy Flavour* in *S1 Density* and *S2 Density* and the *vDGU Strategy Flavour* in *S3 Density* and *S4 Density*.
- Generally speaking, the *S1 Density* and *S2 Density* required higher rates of Agents compared to the *S3 Density* and *S4 Density*. That should be due to the fact that in the Higher Density

Regions, there would be many Users to report sufficient Cases for each Detector. Still, the *Users' Ignorance Rate* did not had a significant influence on the observed results. That is because it is not the number of Cases that matters here. Rather, it is the the accumulative record of the PSPs, MGR records of Section 5.4, that a single victim has dealt with that would matter in the ranking process regardless of the number of the actual reported Cases. Those MGR records are utilised by the powerful TGT Detectors.

- While the *Automatic Strategy Flavour* requires a population of about 50% Active Agents in the *S2 Density*, its requirement drops to less than 20% in the other Density points. That may indicate that the *S2 Density* is within the Threshold Region for the *Automatic Strategy Flavour* and, hence, more Agents are required to detect more suspicious (P)SPs.
- The Agents populations requirements for the *vDGU*, *DGU*, and *Extreme DGU Strategies* are High in the *S1 Density* and *S2 Density* while they would significantly fall down, closer to the *Automatic Strategy Flavour* requirement in the *S3 Density* and *S4 Density*. That may indicate that the *S1 Density* and *S2 Density* are within the Threshold Region for the Strategies that utilise DGU and, hence, they require more Agents to detect more suspicious (P)SPs.
- An alternative possible explanation for the last two points is the fact that the less Users are present, in *S1 Density* and *S2 Density*, the less Cases that would be reported causing the need for the extra Agents.

Milestone 8.3: ANOVA testing the Strategies effects on the *Uncolluding MPSPs Detection Rate*

- **Important Observations:**

- Outside the Threshold Density Regions, where the Density Points *S2* and *S3* are located, there is no advantage to deploy the *Manual Strategy Flavour* instead of the *ST Strategy Flavour* when it comes to improving the *Uncolluding MPSPs Detection Rate*.
- The *Automatic Strategy Flavour* is almost useless when it comes to improving the *Uncolluding MPSPs Detection Rate* in the *S1 Density* and, hence, it should not be deployed in an environment dominated by such Attackers. That might be due to its conservative setting like setting the *Ignore Old G Ranks = true*, *PCR Threshold = 3*, and *Suspicious SP Rank = 20*. Being conservative in a High Malicious Density Region is not a good idea since it would make the Trust situation gets even worse, by the reduced volume of the Detected MPSPs.
- While the *vDGU Strategy Flavour* is significantly better than the *Automatic Strategy Flavour* in the *S2 Density*, it is sometimes similar or even worse than either of the *Automatic* and *ST Strategies*. Given that even the *Automatic Strategy Flavour* is not better than the *ST Strategy Flavour*, if the environment is dominated by Uncolluding MPSPs, there would be no point in deploying either of the *Automatic* or *vDGU Strategies* instead of the *ST Strategy Flavour*.
- The reason for the superior performance of the *DGU Strategy Flavour* compared to the *vDGU Strategy Flavour* in the *S1 Density* and *S2 Density* is the fact that the *vDGU Strategy Flavour* slows down the Detection process by wiping away all the records containing the (P)SPs who are continuously *installing DGU* voluntarily. In the *S1 Density* and the *S2 Density* where the Malicious Density is High, the system can not tolerate slowing down the Detection process.

- **Malicious Density Theory Notes:**

- The *GPD Strategy Flavour* is doing a good job at Detecting *Uncolluding MPSPs* in Density Points where their population is High, i.e. *S1 Density* and *S3 Density*, since they would generate more Cases. However, High *Users' Ignorance Rate* would render this Strategy Flavour useless, because of the reduced reported Cases that are vital for this Strategy Flavour to function well.
- The best Strategy Flavour at Detecting *Uncolluding MPSPs* is definitely the *Extreme DGU*. In all Density Points, its Detection Rate is around 65%. In one case, the *S4 Density*, High *Users' Ignorance Rate* caused an improvement of 5% on the Detection Rate. That could be because this is a Random Strategy Flavour and, hence, less Cases would increase the randomness and the detection as well. Still, the *Banned Innocent PSPs Rates* were unaffected by the *Users' Ignorance Rate*.
- The *ST Strategy Flavour* Detection was stable in all Density Points with values around 20%.
- the *Manual Strategy Flavour* is Detecting best in Density Points *S2 Density* and *S3 Density*. In those Density Points, it achieves significantly more Detection than the *ST Strategy Flavour* by about 10%. However, in Density Points *S1 Density* and *S4 Density*, there was no significant difference between the *ST Strategy Flavour* and the *Manual Strategy Flavour*.
- The Strategies relying on the Weak Colluding Detector, i.e. *Manual*, *vDGU*, *DGU*, in addition to the *ST Strategies*, are a bit sensitive to variations in the *Users' Ignorance Rate*. However, those variations caused significant effects on the *Uncolluding MPSPs Detection Rate* only in Density Points *S4* for the *Manual Strategy Flavour* and the *S1 Density* for the *DGU Strategy Flavour*. That might be explained because those Density Points Region are probably outside the Threshold Region and, hence, variations in the reported Cases numbers would have stronger effects on the Weak Colluding Detector performance. In all cases, the difference was not huge, around 5% decrease in the Detection Rate.
- Since the *Automatic Strategy Flavour* relies mainly on Agents reported Cases, this Strategy Flavour is not significantly affected by variations in the *Users' Ignorance Rate*.
- The *Automatic Strategy Flavour* Detection Threshold Region includes the *S2 Density Point* with around 20% Detection Rate while the *S3 Density* and the *S4 Density* had about 10% and the *S1 Density* had about 5% Detection Rates.
- For the *vDGU Strategy Flavour*, the best Detection Rate is achieved in the *S1 Density* with a Rate of 25% followed by the *S2 Density* with Rate of 20% and then both the *S3 Density* and the *S4 Density* with a rate of 15%. It is noted that the Rate achieved in the *S1 Density* is not significantly different from the Rate achieved in the *S2 Density* while the rate achieved in the *S3 Density* is not significantly different from the Rate achieved in the *S4 Density*. Still, the Rate achieved in the *S1 Density* would be significantly different from the Rate achieved in both the *S3 Density* and *S4 Density*.
- For the *DGU Strategy Flavour*, the best Detection is achieved in the *S1 Density* with a Rate of 45% followed by the *S2 Density* with a rate of 35% and then the *S3 Density* with a Rate of 85% and finally the *S4 Density* with a Rate of 95%. All of those Rates are slightly significant compared to each other.
- Putting the *GPD* and *Extreme DGU Strategies* aside, due to their unbearable side effects, the *DGU Strategy Flavour* is significantly the best *Uncolluding MPSPs* Detector in the *S1 Density*. Moreover, it is significantly better than the *vDGU Strategy Flavour* and comparable to *Manual Strategy Flavour* in the *S2 Density*. However, it is the worst Detector in the *S3 Density* and the

S4 Density with comparable performance to the *vDGU* and *Automatic Strategies*. It should be noted that this Strategy Flavour is sensitive to variations in the *Users' Ignorance Rate* due to its dependence on the Weak Colluding Detector.

Milestone 8.4: ANOVA testing the Strategies effects on the *Uncolluding MSPs Detection Rate*

- **General Notes:**

- Due to their low activity profile, the *GPD Strategy Flavour* is unable to detect Uncolluding MSPs under any environmental scenarios.
- Since only the *ST* and *Manual Strategies* deploy the *Top TLRavg ST Selector*, these Strategies are the only effective ones at achieving significant *Uncolluding MSPs Detection Rates*. However, this achievement is at the cost of requiring a *High Active Agents to Users Rate*.

- **Malicious Density Theory Notes:**

- The *Uncolluding MSPs Detection Rates* for both of the *ST* and *Manual Strategies* are around 15% without any significant difference in most Regions and ignorance rates except for:
 - * The *S2 Density* and *Low Users' Ignorance Rate*: The *ST Strategy Flavour* performed better than the *Manual Strategy Flavour* by about 5%.
 - * The *S3 Density*: The *Manual Strategy Flavour* performed significantly worse than itself when the *Users' Ignorance Rate* was set High. In addition, the *Manual Strategy Flavour* did significantly worse than the *ST Strategy Flavour* when the *Users' Ignorance Rate* was set High.
- The reason why the *Manual Strategy Flavour* is affected by variations in the *Users' Ignorance Rate* could be the fact that the *ST Strategy Flavour* Selectors are dependant on the TG Rankings which are affected mainly by the W_{TGT} . In the other hand, the *ST Strategy Flavour* performance is mainly affected by the settings of the W_{TST} and W_{TLRavg} . When the malicious population is low in the *S2 Density* and the *S3 Density*, MSPs may not get noticeable with higher than normal ranks, due to the low volume of interactions they would be able to engage in. If they happen to have lower ranks, they would get caught by the *Bottom TLRavg ST Selector* which is utilised only by the *ST Strategy Flavour*.

Milestone 8.5: ANOVA testing the Strategies effects on the *Colluding Popular MPSPs Detection Rate*

- **General Notes:**

- Since the Colluding MPSPs Attacks wont deploy any colluding against the trivial *GPD Strategy Flavour*, this Strategies performance at Detecting Colluding Popular MPSPs is similar to its' performance at Detecting Uncolluding MPSPs 8.3.
- With the exception of the *GPD Strategy Flavour*, the *Users' Ignorance Rate* did not generate any significant difference in the Strategies performance at Detecting the Colluding Popular MPSPs.
- Since the *ST Strategy Flavour* can not Detect Colluding MPSPs, and given that the Colluding MPSPs detected by the Strategies utilising the DGU would adopt Advanced Colluding Settings that would make their Detection almost impossible, the *ST Strategy Flavour* and the DGU variants of Strategies have negligible effects on the *Colluding Popular MPSPs Detection Rate*.

- **Malicious Density Theory Notes:**

- The Detection Rates achieved by the *Manual Strategy Flavour* are around 22% in the *S1 Density*, 35% in the *S2 Density*, 30% in the *S3 Density*, and 10% in the *S4 Density*. That is going in confirmation with the predictions of the Malicious Density Theory about the Threshold Region.
- The Detection Rates of the *Automatic Strategy Flavour* are around 0% in the *S1 Density*, 25% in the *S2 Density*, 10% in the *S3 Density*, and 15% in the *S4 Density*. That is an unexpected diversion from the predictions of the Malicious Density Theory. It looks like that this Strategy Flavour works better with Low Malicious population and, hence, the sharp divergence when the Malicious Population get High. This divergence goes to the point that even increasing the Normal population wont be as effective at improving the Detection Rates as it would normally do with the rest of the Strategies. That might be due to the fact that this Strategy Flavour is ultra conservative which is not a good quality in High Malicious Density Regions.

Milestone 8.6: ANOVA testing the Strategies effects on the *Colluding Unpopular MPSPs Detection Rate*

- **General Notes:**

- Given how complicated this attack is, the only Strategy Flavour that would be able to Detect a significant number of these Attackers is the *Manual Strategy Flavour* due to its reliance on the Weak Colluding Detector. Despite that all the DGU variants of Strategies are also dependant of the Weak Colluding Detector, they tend to give Detected Colluding MPSPs a second life where they would *Install DGU* and, then, adopt Advanced Colluding Settings making their Detection almost impossible.

- **Malicious Density Theory Notes:**

- It is noted that the Threshold Regions for the *Manual Strategy Flavour* include the *S2 Density* followed by the *S3 Density* while its worst Detection Region would include the *S4 Density* where the Detection is almost vanished as predicted by the Density Theory. It is also noted that when the *Users' Ignorance Rate* is High, the Detection Rate would decrease by a range between 5% and 10%. That is expected since the Weak Colluding Detector, which this Strategy Flavour relies on, is dependant on Users Cases. Still, only the differences in the *S1 Density* and the *S3 Density* are slightly significant.

Milestone 8.7: ANOVA testing the Strategies effects on the *Compromised Share with Top Credentials Rate*

- **Important Observations:**

- It is noted that the *GPD Strategy Flavour* is doing almost like the *None Strategy Flavour* despite the High Detection Rates it achieves in some Density Points. That might be because of the inaccurate TG Rankings, as a consequence of the Attackers' utilisation of Simple Attacking Strategies 5.8.6, leading most Users to deal with the most Malicious MPSPs given their High TG Ranks. That is also true for the rest of the Strategies but the Attackers wont dare abusing the Credentials they own unless they satisfy some complex Colluding Settings to avoid Detection by the rest of the smarter Strategies.

- It is also noted that when the *Users' Ignorance* is High, the *GPD Strategy Flavour* is doing much worse than the *None Strategy Flavour* due to the above mentioned issue of inaccurate TG Rankings combined with the fact that the *GP Strategy Flavour* can not Detect well without a large supply of reported Cases which would lead to a surge of the Malicious Nodes in the network.

- **General Notes:**

- It is noted that the *Users' Ignorance Rate* does not have significant effects on any of the Strategies' achieved *Compromised Share with Top Credentials Rates*, apart of the *None* and *GPD Strategies*.
- The *Compromised Share with Top Credentials Rate* achieved by the *ST Strategy Flavour* is constant around 10% in all Density Points.
-

- **Malicious Density Theory Notes:**

- The *Compromised Share with Top Credentials Rate* achieved by the *Manual Strategy Flavour* would be between 15% and 20% in the *S1 Density* and the *S2 Density* while it would significantly drops to around 10% in the *S3 Density* and the *S4 Density*. The reason for the High rate of the *Compromised Share with Top Credentials* achieved by this Strategy Flavour could be the fact that its Counter-Attack setting is to generate more abuse in order to confuse it. More abuse would cause the High Compromising Rates. In Low Malicious Density settings, the increased abuse effects would get negligible.
- The *Automatic Strategy Flavour* achieved similar results of the *Manual Strategy Flavour* except in the *S1 Density* and the *S4 Density* where it achieved slightly better results. Given the Counter-Attack settings against both Strategies are the same and that the *Manual Strategy Flavour* is doing better at Detecting M(P)SPs, the explanation to the *Automatic Strategy Flavour* superiority in reducing the Compromising Rate would be the reduced *Users Trust* that is caused by its poor Detection Rates leading to reduced traffic in the network and, hence, reduced Compromising Rates.
- The *DGU Strategy Flavour* achieved similar results of the *Automatic Strategy Flavour* except in case of High *Users' Ignorance Rate* where it achieved slightly better results.
- The *DGU Strategy Flavour* performed almost exactly like *Automatic Strategy Flavour*. Maybe it is the extra speed of Detection, compared to the *vDGU Strategy Flavour*, that would enable the *DGU Strategy Flavour* to achieve a similar reduced Compromising Rates of the more conservative *Automatic Strategy Flavour*.
- The *Extreme DGU Strategy Flavour* has a similar effect of the *ST Strategy Flavour* except in *S4 Density* where it is doing slightly better and in the *S1 Density* with a High *Users' Ignorance Rate* where it is doing slightly worse. The good performance of the *Extreme DGU Strategy Flavour* might be caused by the Advanced Colluding Settings that is adopted by its Counter-Attackers leading to less overall abusing activity. In the *S4 Density* where the Malicious Nodes population is Low, it would be even harder to satisfy the strict colluding conditions leading to even better performance of the *Extreme DGU Strategy Flavour*.

Milestone 8.8: ANOVA testing the Strategies effects on the *Compromised Share with Any Credentials Rate*

- **Important Observations:**

- It is noted that the *GPD Strategy Flavour* is doing almost like the *None Strategy Flavour*, just like its performance discussed in 8.7.

- **General Notes:**

- It is noted that the *Users' Ignorance Rate* does not have significant effects on any of the Strategies' achieved *Compromised Share with Any Credentials Rates*, apart of the *None* and *GPD Strategies*.
- The *Compromised Share with Any Credentials Rate* achieved by the *ST Strategy Flavour* is constant around 5% in all Density Points.
-

- **Malicious Density Theory Notes:**

- The *GPD Strategy Flavour* has the same *Compromised Share with Any Credentials Rate* as the *None Strategy Flavour* except in the *S2 Density* and the *S3 Density*, where the Malicious Density is low. The Detection Rates in these Density Points are still better than the *None Strategy Flavour* and the *Banned Innocent PSPs* does not reach more than 5%. So, a possible explanation for the High *Compromised Share with Any Credentials Rate* could be the fact that the more you detect, the smaller the Malicious Density gets making it even harder to detect the M(P)SPs hiding among piles on Innocent (P)SPs. Instead, they would gain High Ranks causing them to get even more traffic of credentials.
- The *Compromised Share with Any Credentials Rate* achieved by the *Manual Strategy Flavour* would be between 10% and 15% in the *S1 Density* and the *S2 Density* while it would significantly drops to a value between 5% and 10% in the *S3 Density* and the *S4 Density*. The reason for the High rate of the *Compromised Share with Top Credentials* achieved by this Strategy Flavour could be the fact that its Counter-Attack setting is to generate more abuse in order to confuse it. More abuse would cause the High Compromising Rates. In Low Malicious Density settings, the increased abuse effects would get negligible.
- Unlike the situation in 8.7, The *Automatic Strategy Flavour* achieved similar performance to the *Manual Strategy Flavour* even in the *S1 Density* and the *S4 Density*. The reason might be the fact that the *Share with Top Credentials* would be biased for sharing with (M)PSPs. Since the MPSPs have High Density in *S1 Density* and the *S4 Density* if the *Manual Strategy Flavour* is deployed because its achieved *Banned Innocent PSPs Rates* in those Density Points are High while the *MPSPs Detection Rates* are Low.
- Both the *DGU Strategy Flavour* and the *vDGU Strategy Flavour* performed similarly good and slightly significantly better than the *ST Strategy Flavour* except when the *Users' Ignorance Rate* is Low in the *S2 Density* and the *S4 Density*. The better performance can not be explained simply by the better *Uncolluding MPSPs Detection Rates* that are achieved by the DGU variants Strategies in the *S1 Density* and the *S2 Density*. That is because even the *ST Strategy Flavour* would achieve better Detection Rates in *S3 Density* and the *S4 Density* in addition to its considerably better *Uncolluding MSPs Detection Rate* compared to the DGU variants of Strategies. A better explanation would be the fact that the DGU variants of Strategies would detect a much larger population of (M)PSPs and, then, give them the option to *Install DGU* where they would only be allowed to share the Credentials they acquire with only trusted (P)SPs, who have also

installed DGU, or PSPs. Sharing to PSPs would create a bias toward abusing the *Share with Top Credentials*. In addition, When the Malicious population is Low, the *ST Strategy Flavour* would be able to catch up unless the *Users' Ignorance Rate* is High where it wont be as effective as the DGU variants Strategies.

- The *Extreme DGU Strategy Flavour* has a slightly significantly better effect compared to the *ST Strategy Flavour* except in the *S4 Density* where the *Users' Ignorance Rate* is Low as their effects where similar in that particular situation. The good performance of the *Extreme DGU Strategy Flavour* might be caused by the fact that the main contributor to abusing the *Share with Any Credentials* would be the *Uncolluding MPSPs* which are heavily Detected by the *Extreme DGU Strategy Flavour*.

Milestone 8.9: ANOVA testing the Strategies effects on the *Banned Innocent PSPs "Guilty" Rate*

- **General Notes:**

- All Defensive Strategies achieved *Banned Innocent PSPs "Guilty" Rates* were not significantly different from 0 except for the *GPD Strategy Flavour* and the *Manual Strategy Flavour* regardless of the level of the *Users' Ignorance Rate*.
- When it comes to the *Manual Strategy Flavour*, High *Users' Ignorance Rate* would lead to an improvement decrease in the Banning Rates of about 2%. That might be because the less reported Cases and the longer Cases, due to the delay in reporting them, would increase the *Manual Strategy Flavour* uncertainties forcing it to slow down in the Banning process.

- **Malicious Density Theory Notes:**

- It is noted that in the *S4 Density*, the *GPD Strategy Flavour* would ban about 3% Innocent PSPs while in the *S3 Density*, it would ban about 5%. That is, in Regions of High Normal Nodes Density, the *GPD Strategy Flavour* would ban more Innocent PSPs. That could be because in all Regions, the rankings are distorted by the Simple Attacking Strategies of 5.8.6 and the increase of reported Cases has simply magnified the distortion and make the Banning mistakes visible within the simulation period. i.e. in longer simulations, it is expected that all Density Regions would show similar Banning Rates if not even more. The fact that High *Users' Ignorance Rate* leads to a decrease in the Banning Rates supports this explanation.
- the *Manual Strategy Flavour* falsely Banned about 7% Innocents in the *S1 Density*, 26% in the *S2 Density*, 18% in the *S3 Density*, and 13% in the *S4 Density*. The reason for the extraordinary High Banning Rate would be the fact that this is an aggressive Random Detector. The massive increase in the *S4 Density* followed by the *S3 Density* is explained by the Malicious Density Theory 6.8.

Milestone 8.10: ANOVA testing the Strategies effects on the *Banned Innocent PSPs "Refused to Install DGU" Rate*

- **Important Observations:**

- This measure is only applicable to DGU variants of the Defensive Strategies.
- The *vDGU Strategy Flavour* achieved almost 0% *Banned Innocent PSPs "Refused to Install DGU" Rate* in all the Density Points since most Innocent PSPs would voluntarily *Install DGU*.

- The *Extreme DGU Strategy Flavour* is significantly the worst offender followed by the *DGU* who still generated High Banning Rates.

- **Malicious Density Theory Notes:**

- In the *S1 Density*, the *Extreme DGU Strategy Flavour* achieved about 22% *Banned Innocent PSPs* while the *DGU Strategy Flavour* achieved about 13%. When the *Users' Ignorance Rate* is High, the Banning Rates changed dramatically to 15% for the *Extreme DGU Strategy Flavour* and 10% for the *DGU Strategy Flavour*.
- In the *S2 Density*, the *Extreme DGU Strategy Flavour* achieved about 14% *Banned Innocent PSPs* while the *DGU Strategy Flavour* achieved about 8%. When the *Users' Ignorance Rate* is High, the Banning Rates were almost the same.
- In the *S3 Density*, the *Extreme DGU Strategy Flavour* achieved about 17% *Banned Innocent PSPs* while the *DGU Strategy Flavour* achieved about 2%. When the *Users' Ignorance Rate* is High, the Banning Rates were almost the same.
- In the *S1 Density*, the *Extreme DGU Strategy Flavour* achieved about 9% *Banned Innocent PSPs* while the *DGU Strategy Flavour* achieved about 1%. When the *Users' Ignorance Rate* is High, the Banning Rates were almost the same.
- Since the *Extreme DGU Strategy Flavour* is dependant on *Users' reported Cases*, the *Users' Ignorance Rate* have huge impact on its integrity. That is true in High Density Regions where this Strategy Flavour is confused due to the enormous amount of abuse Cases it receives. The less Cases it gets, the less accusations in general, including banning decisions, it would make. the *DGU Strategy Flavour* is also dependant on *Users' reported Cases* but it also utilises *Agents feedback* making it more conservative in comparison to the *Extreme DGU Strategy Flavour*.
- In Low Density Points, things are more settled down and there is almost no sensitivity towards variations in the *Users' Ignorance Rate*.

In Stage 6.6.8, we distinguished between 7 different Defensive Strategies that the Auditor could deploy. We compared, by means of ANOVA testing, the performance of those 7 Strategies against the case where the Auditor would simply do nothing. Below is a selected set of our ANOVA comparisons graphs for some of the most relevant responses. Note that we have repeated the comparison at different population settings as well as we varied the *Users Ignorance Rate* Factor which simply shows how sensitive genuine Users would be against the received Abuse. So, the title of each figure would start with the measured response followed by the value of the grouped setting of the *ini Users & (P)SPs* and the *Users & (P)SPs GR*, which could be either Low or High. Then, the value of the grouped setting of the *ini M(P)SPs* and the *M(P)SPs GR* follows in a similar fashion. Finally, the value of the *Ignorance Rate* factor, which could be either 20% or 80%, follows. Another note to consider is the fact that unlike our analysis in Chapter 6 where we analysed the responses of *MPSPs Detection*

Rates, we show in the below figures the opposite response, *MPSPs Undetection Rates*. The reason is simple that during the simulation process, the *MPSPs Undetection Rates* were utilised but we decided to just use its inverse in this Thesis to make our conclusions clear and more understandable.

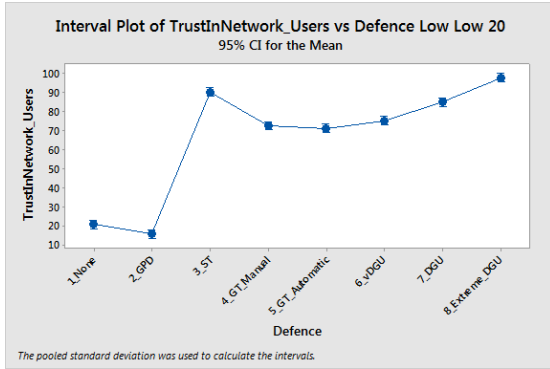


FIGURE C.1: Trust Low Low 20

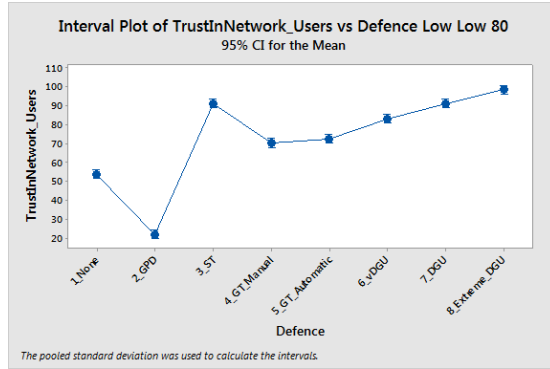


FIGURE C.2: Trust Low Low 80

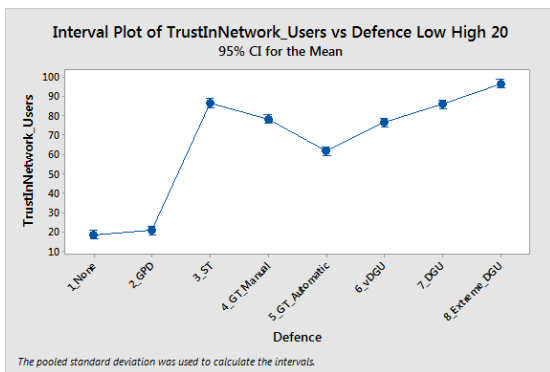


FIGURE C.3: Trust Low High 20

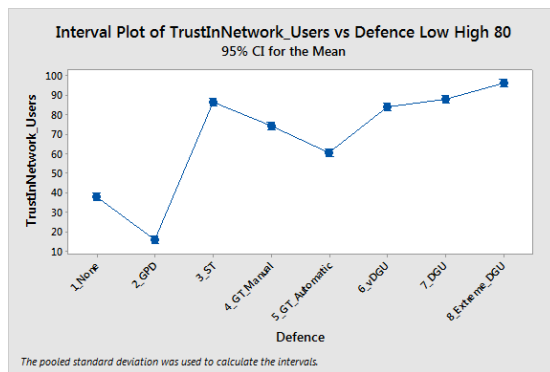


FIGURE C.4: Trust Low High 80

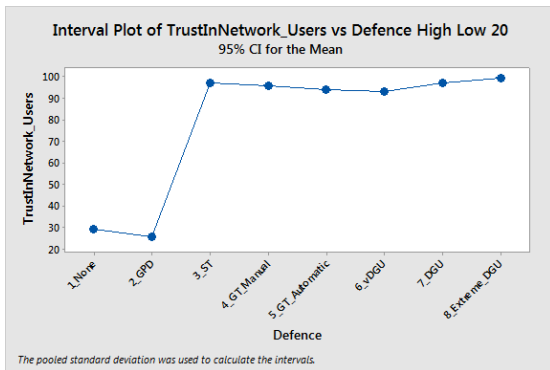


FIGURE C.5: Trust High Low 20

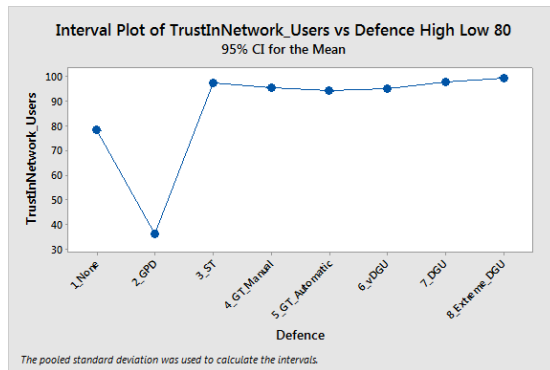


FIGURE C.6: Trust High Low 80

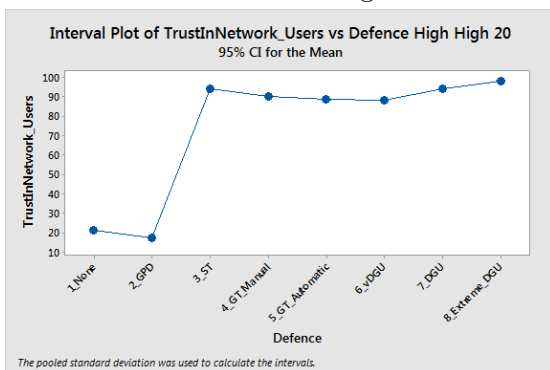


FIGURE C.7: Trust High High 20

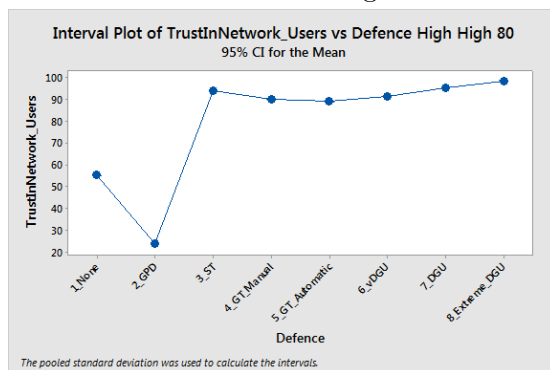


FIGURE C.8: Trust High High 80

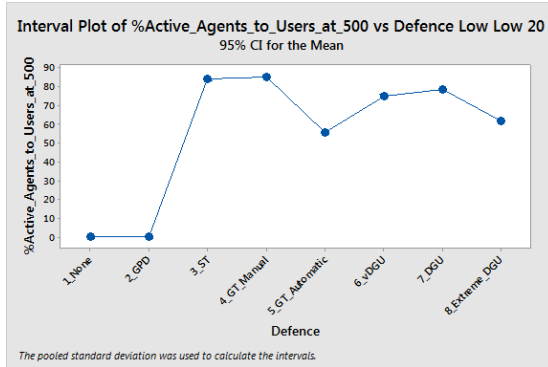


FIGURE C.9: Agents Low Low 20

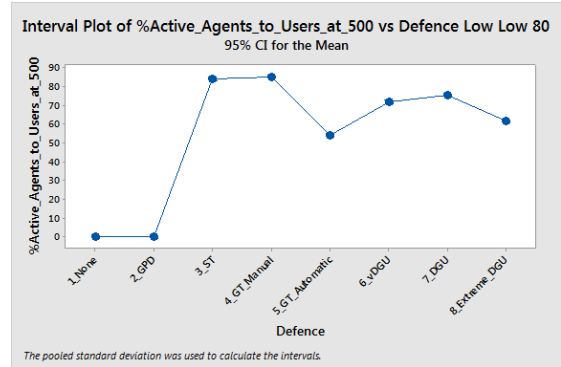


FIGURE C.10: Agents Low Low 80

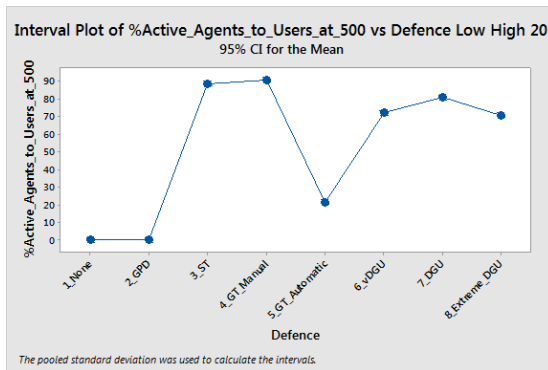


FIGURE C.11: Agents Low High 20

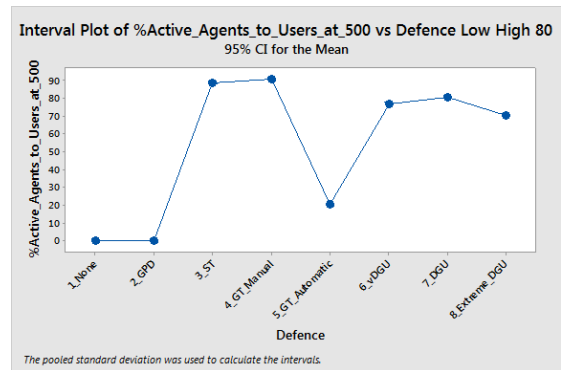


FIGURE C.12: Agents Low High 80

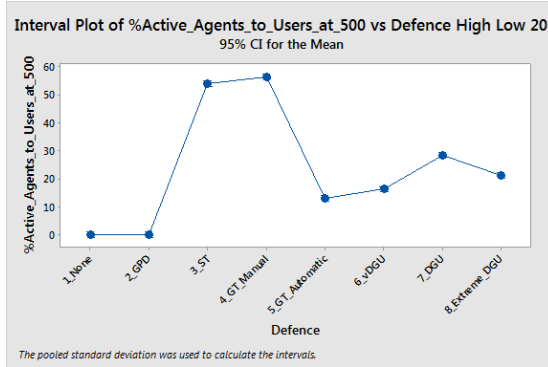


FIGURE C.13: Agents High Low 20

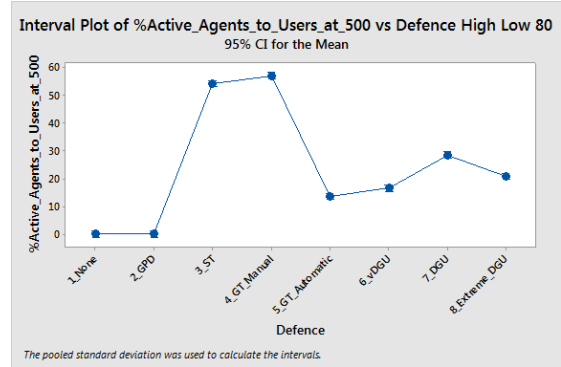


FIGURE C.14: Agents High Low 80

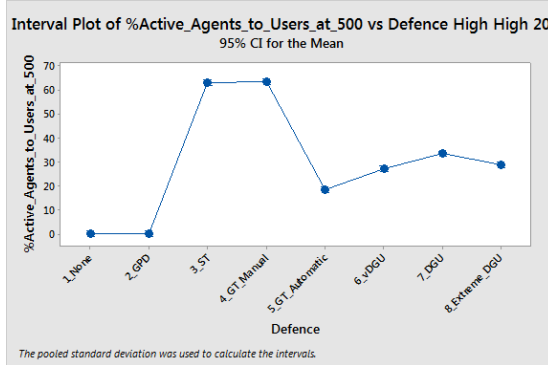


FIGURE C.15: Agents High High 20

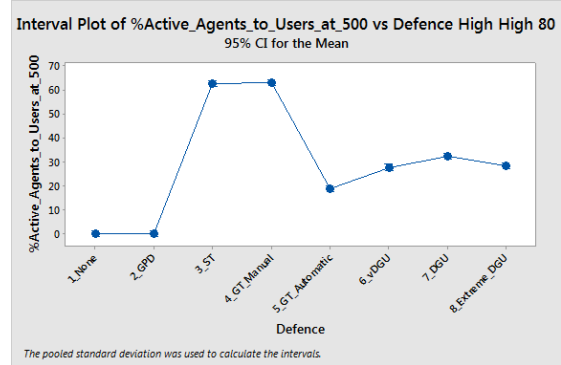


FIGURE C.16: Agents High High 80

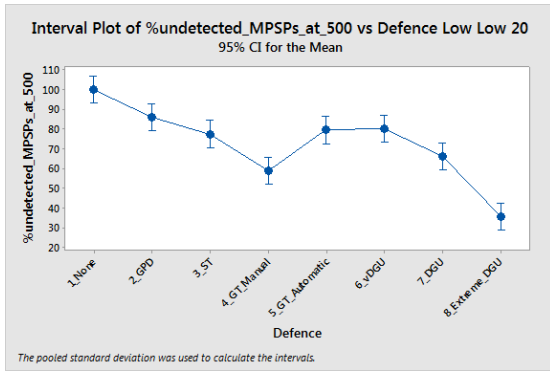


FIGURE C.17: Undetected MPSPs Low Low 20

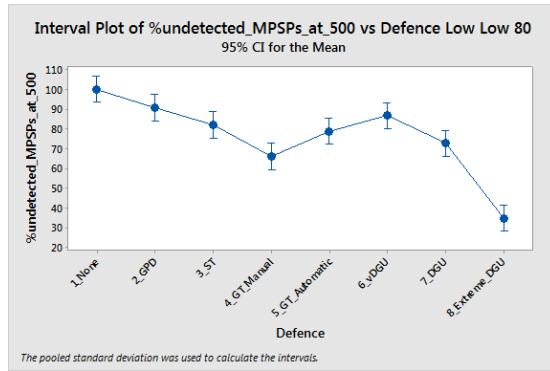


FIGURE C.18: Undetected MPSPs Low Low 80

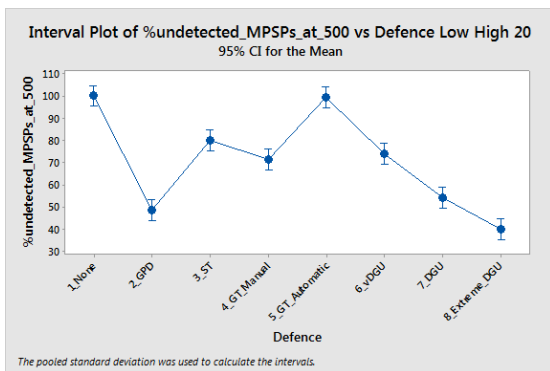


FIGURE C.19: Undetected MPSPs Low High 20

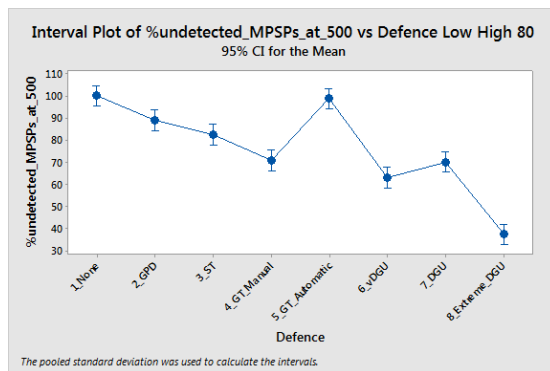


FIGURE C.20: Undetected MPSPs Low High 80

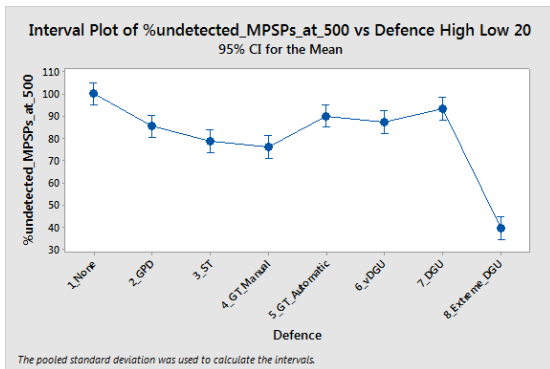


FIGURE C.21: Undetected MPSPs High Low 20

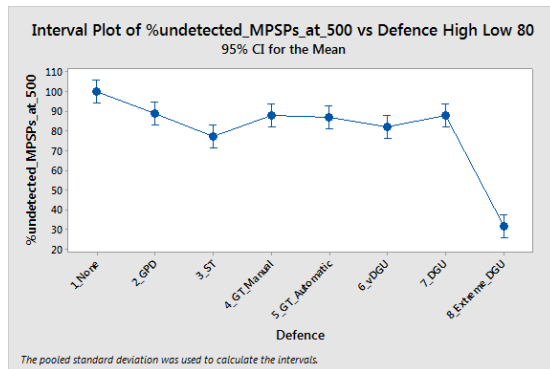


FIGURE C.22: Undetected MPSPs High Low 80

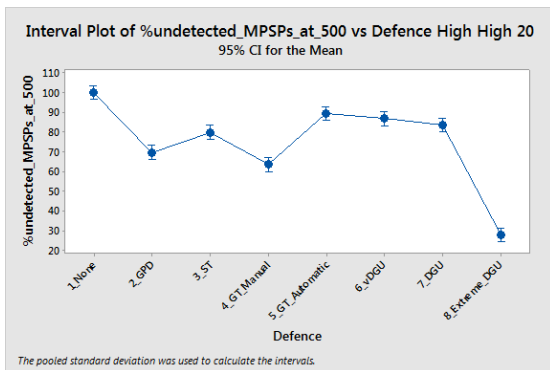


FIGURE C.23: Undetected MPSPs High High 20

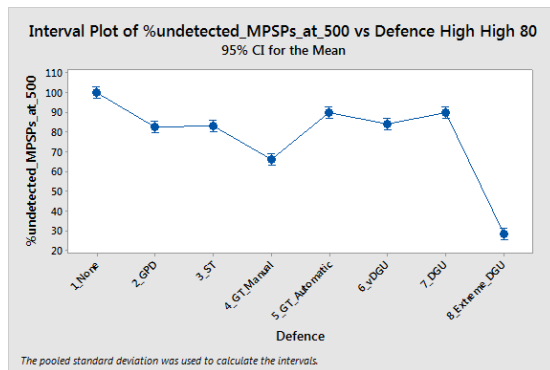


FIGURE C.24: Undetected MPSPs High High 80

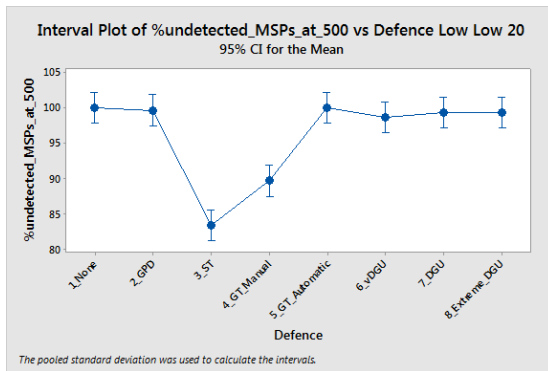


FIGURE C.25: Undetected MSPs Low Low 20

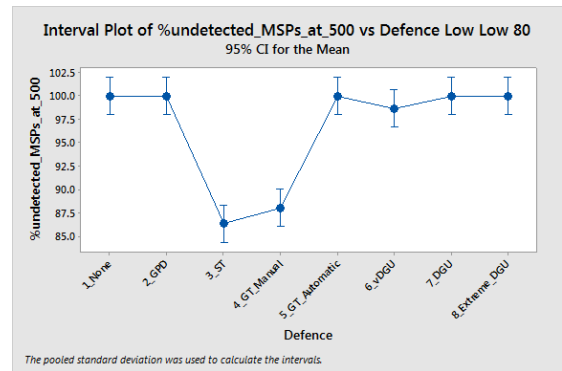


FIGURE C.26: Undetected MSPs Low Low 80

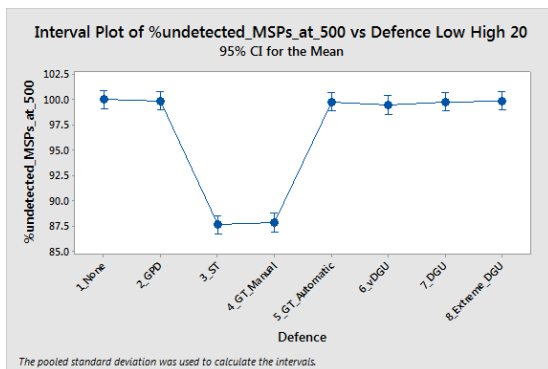


FIGURE C.27: Undetected MSPs Low High 20

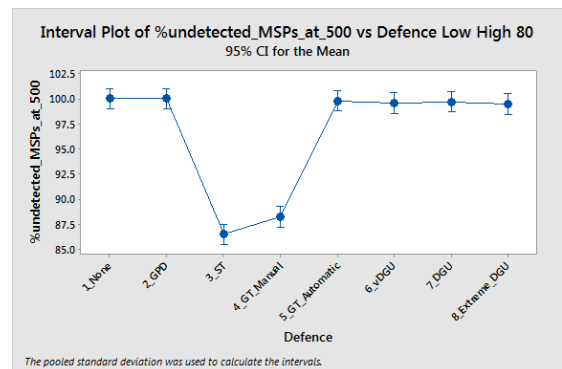


FIGURE C.28: Undetected MSPs Low High 80

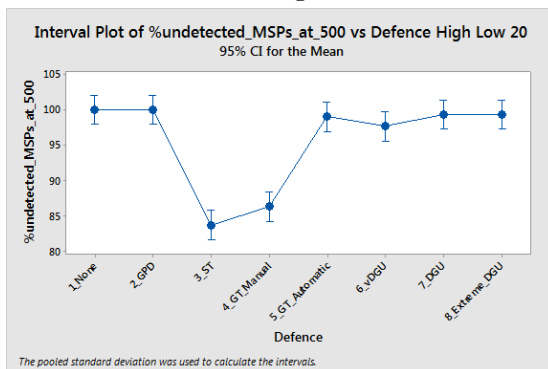


FIGURE C.29: Undetected MSPs High Low 20

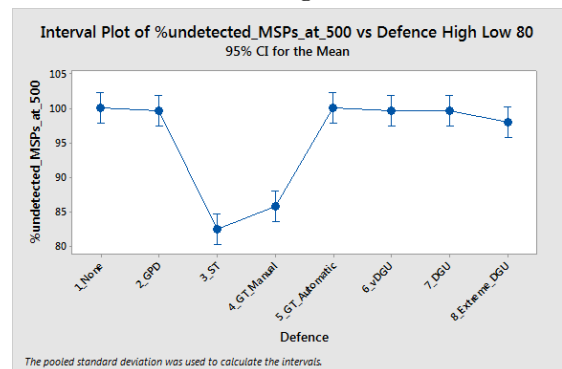


FIGURE C.30: Undetected MSPs High Low 80

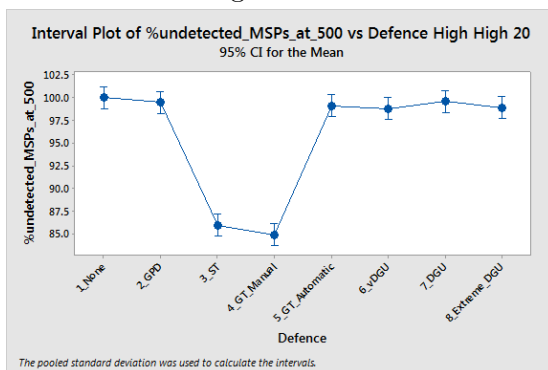


FIGURE C.31: Undetected MSPs High High 20

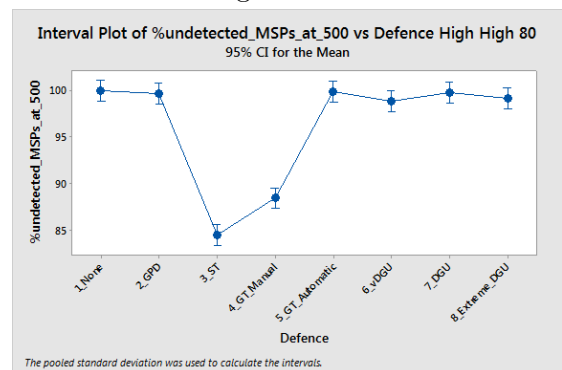


FIGURE C.32: Undetected MSPs High High 80

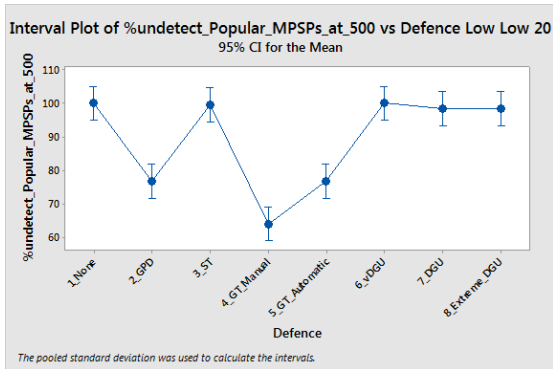


FIGURE C.33: Undetected Popular MPSPs Low Low 20

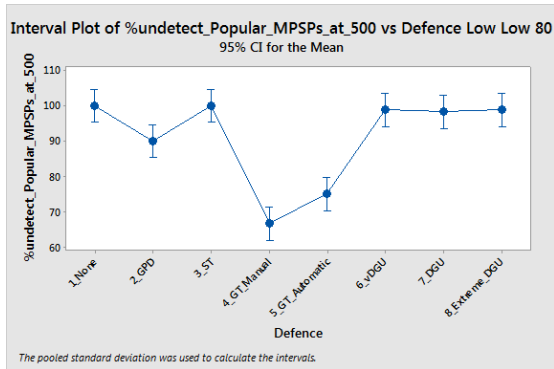


FIGURE C.34: Undetected Popular MPSPs Low Low 80

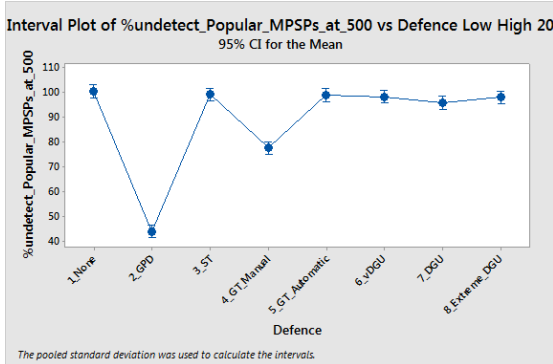


FIGURE C.35: Undetected Popular MPSPs Low High 20

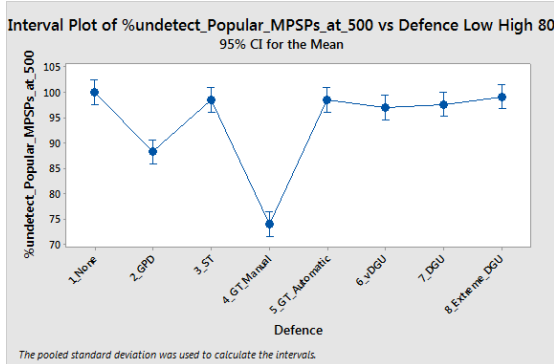


FIGURE C.36: Undetected Popular MPSPs Low High 80

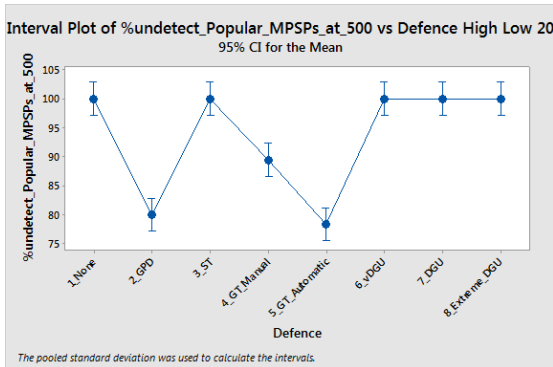


FIGURE C.37: Undetected Popular MPSPs High Low 20

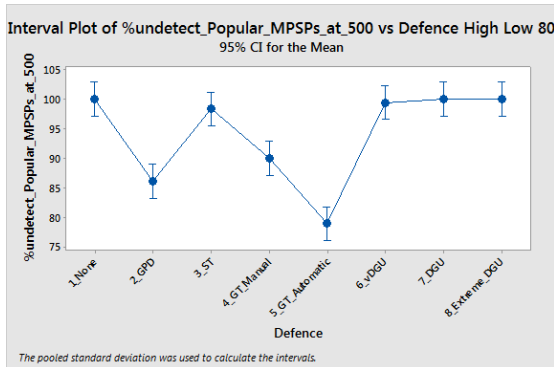


FIGURE C.38: Undetected Popular MPSPs High Low 80

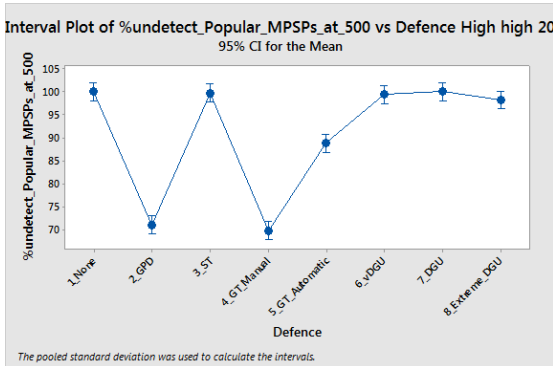


FIGURE C.39: Undetected Popular MPSPs High High 20

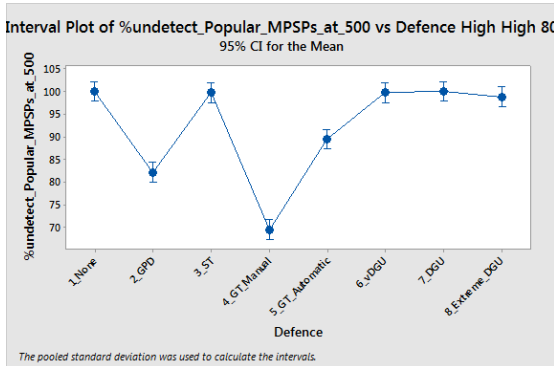


FIGURE C.40: Undetected Popular MPSPs High High 80

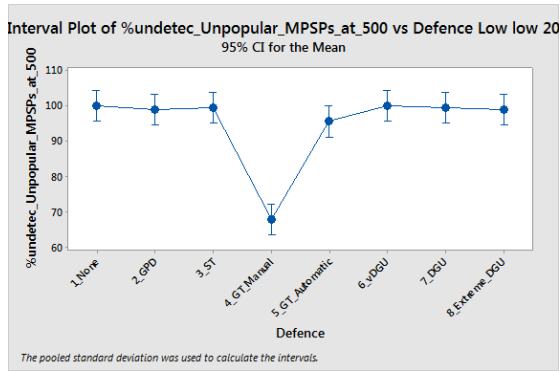


FIGURE C.41: Undetected Unpopular MPSPs Low Low 20

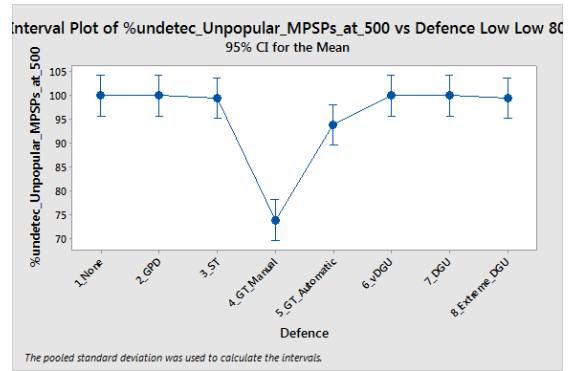


FIGURE C.42: Undetected Unpopular MPSPs Low Low 80

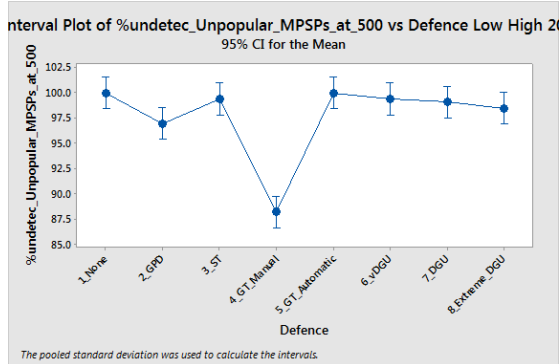


FIGURE C.43: Undetected Unpopular MPSPs Low High 20

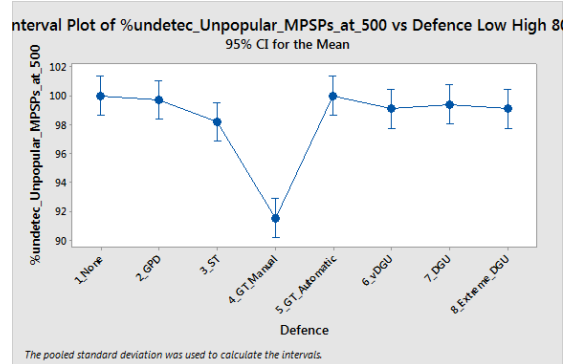


FIGURE C.44: Undetected Unpopular MPSPs Low High 80

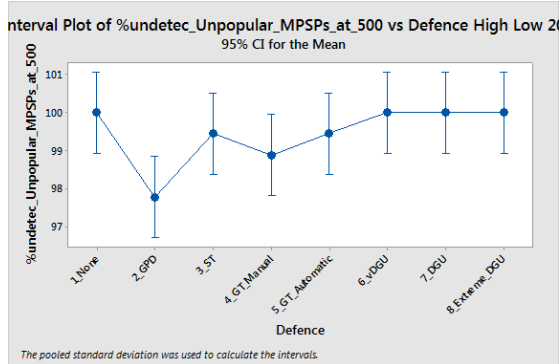


FIGURE C.45: Undetected Unpopular MPSPs High Low 20

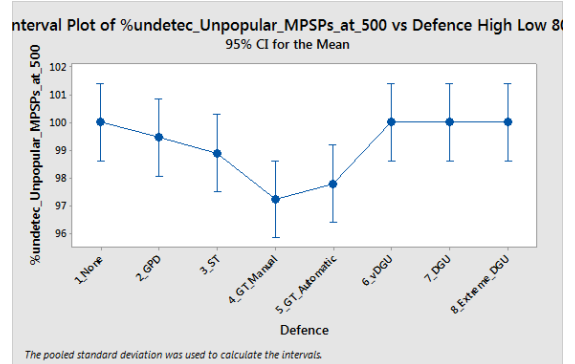


FIGURE C.46: Undetected Unpopular MPSPs High Low 80

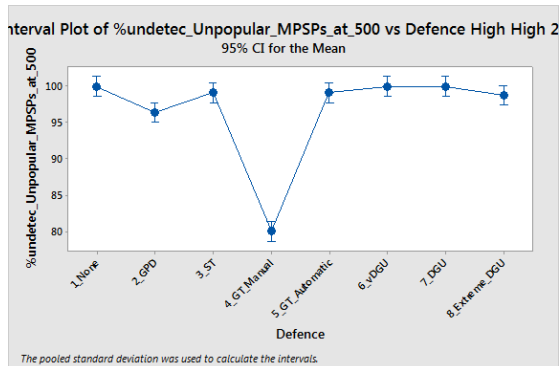


FIGURE C.47: Undetected Unpopular MPSPs High High 20

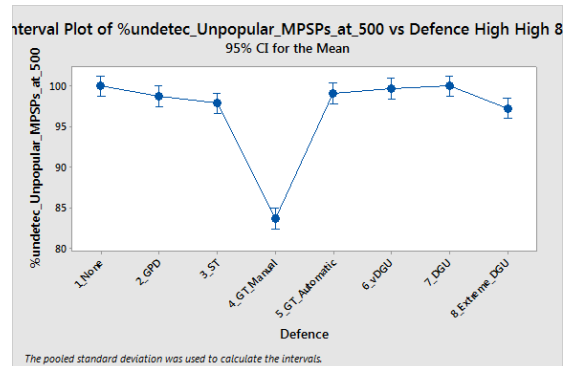


FIGURE C.48: Undetected Unpopular MPSPs High High 80

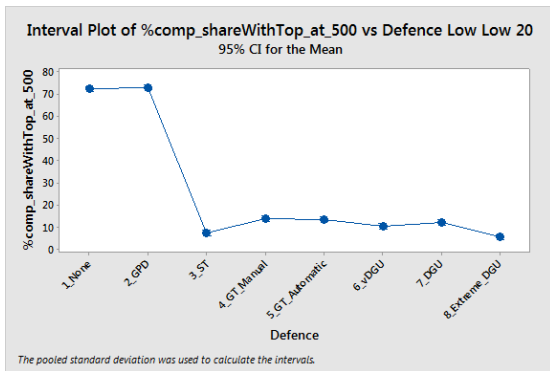


FIGURE C.49: Compromised share-TopCred Low Low 20

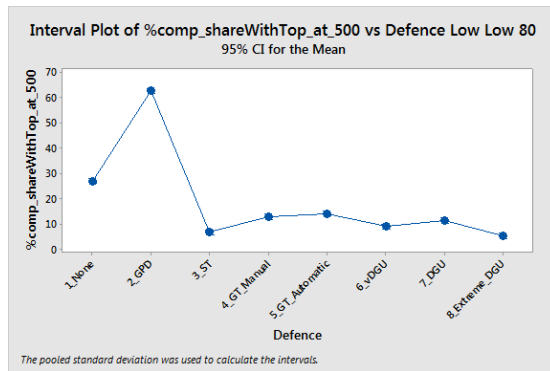


FIGURE C.50: Compromised share-TopCred Low Low 80

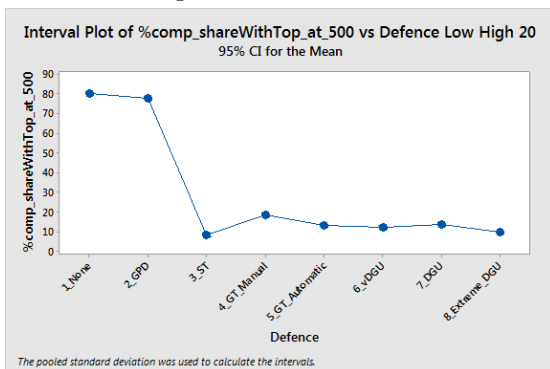


FIGURE C.51: Compromised share-TopCred Low High 20

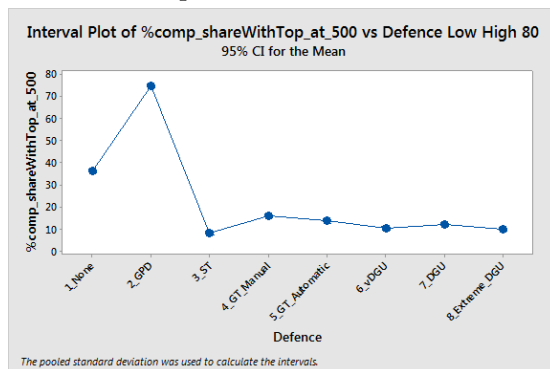


FIGURE C.52: Compromised share-TopCred Low High 80

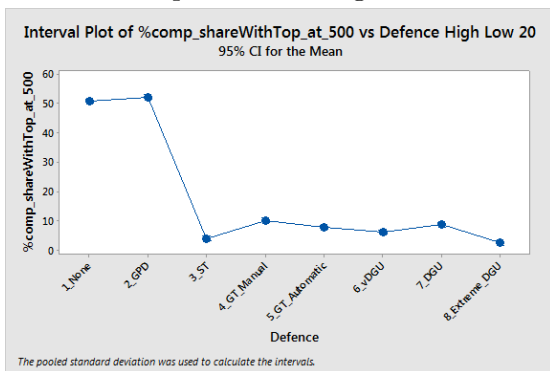


FIGURE C.53: Compromised share-TopCred High Low 20

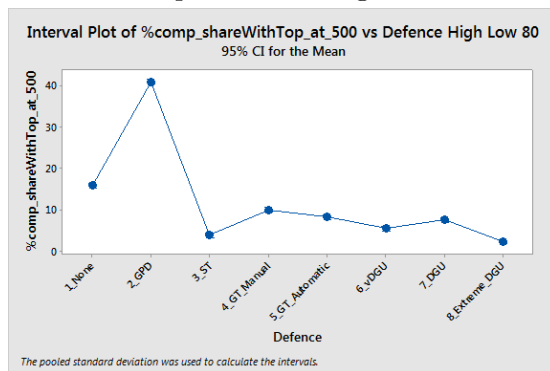


FIGURE C.54: Compromised share-TopCred High Low 80

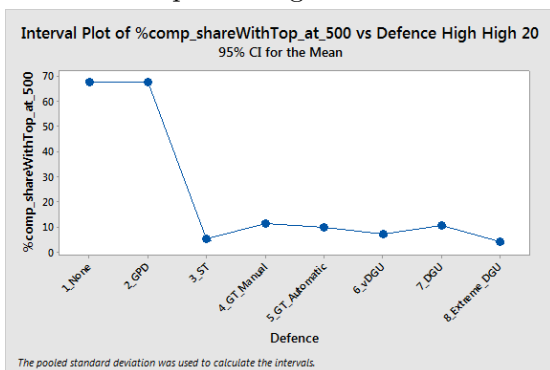


FIGURE C.55: Compromised share-TopCred High High 20

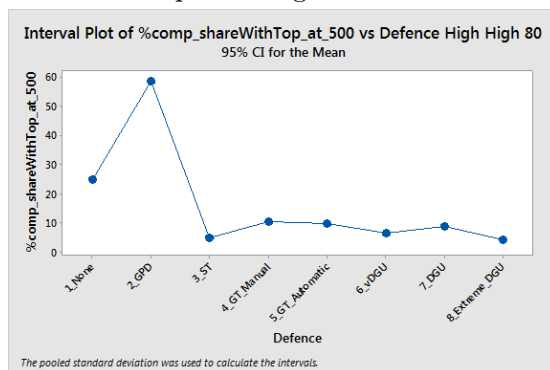


FIGURE C.56: Compromised share-TopCred High High 80

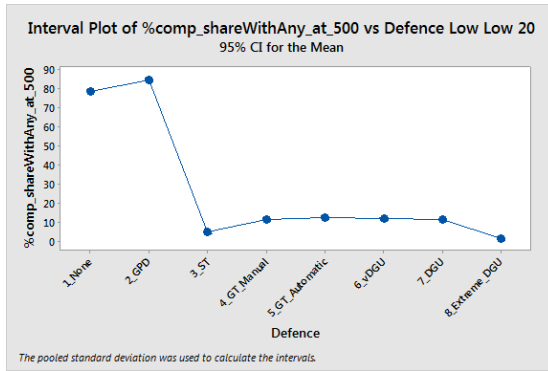


FIGURE C.57: Compromised share-AnyCred Low Low 20

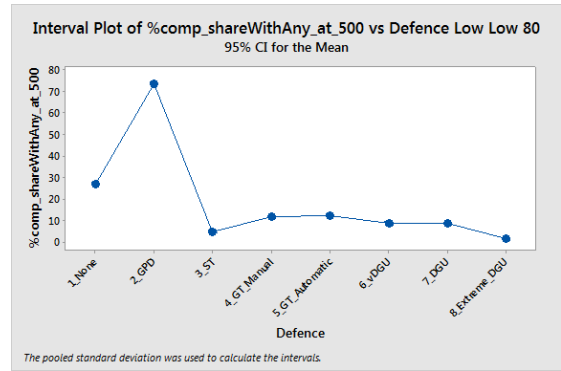


FIGURE C.58: Compromised share-AnyCred Low Low 80

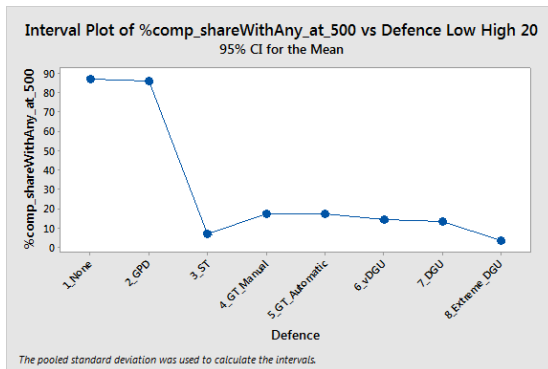


FIGURE C.59: Compromised share-AnyCred Low High 20

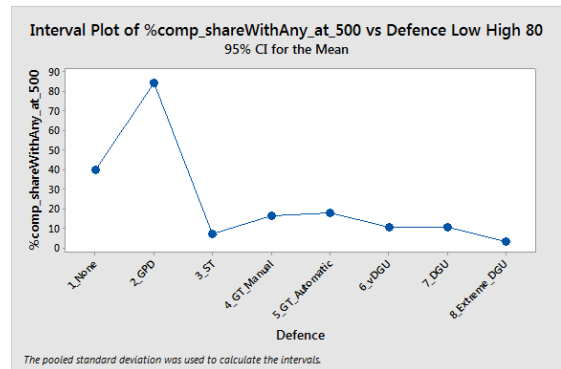


FIGURE C.60: Compromised share-AnyCred Low High 80

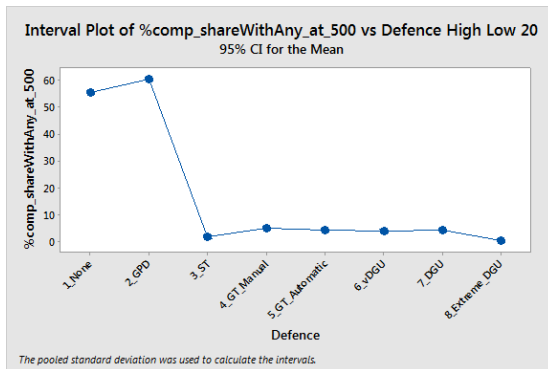


FIGURE C.61: Compromised share-AnyCred High Low 20

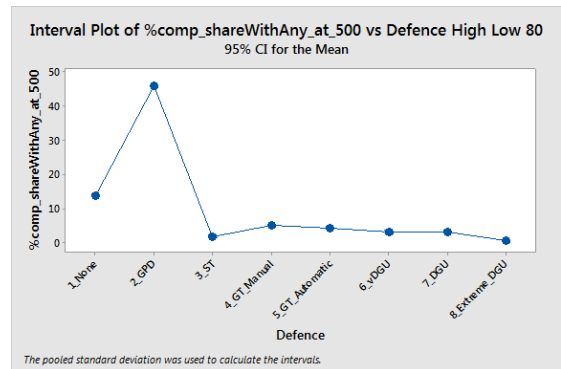


FIGURE C.62: Compromised share-AnyCred High Low 80

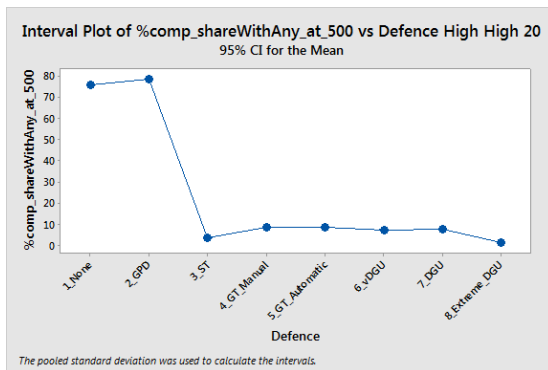


FIGURE C.63: Compromised share-AnyCred High High 20

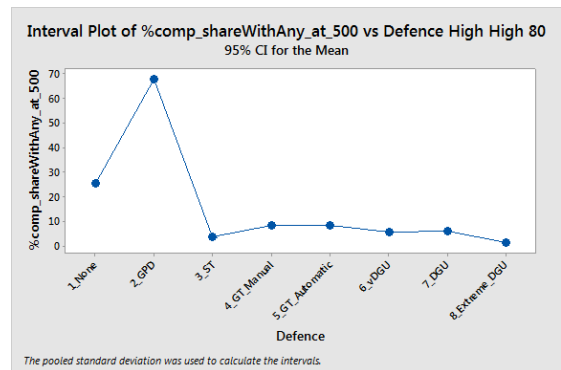


FIGURE C.64: Compromised share-AnyCred High High 80

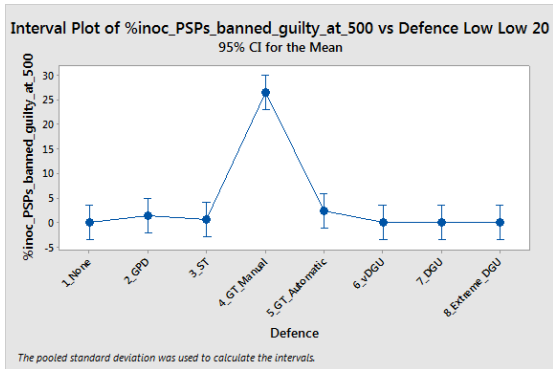


FIGURE C.65: BannedGuilty Low Low 20

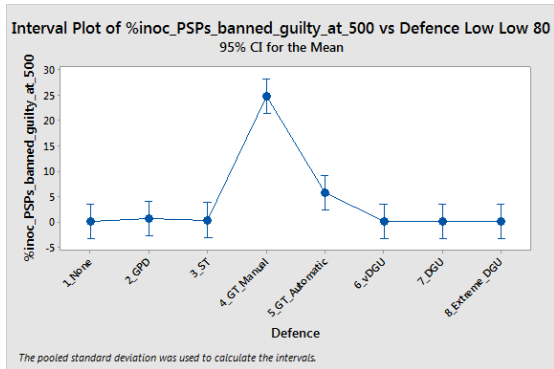


FIGURE C.66: BannedGuilty Low Low 80

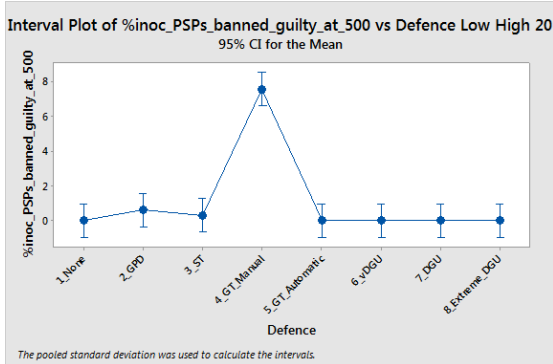


FIGURE C.67: BannedGuilty Low High 20

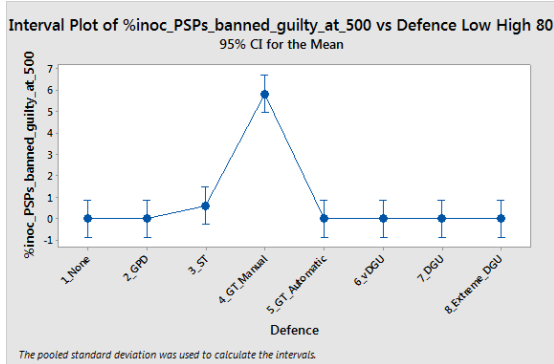


FIGURE C.68: BannedGuilty Low High 80

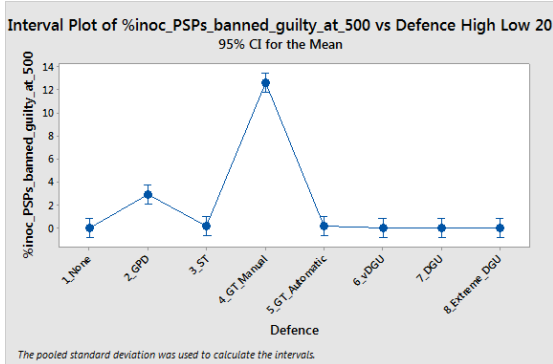


FIGURE C.69: BannedGuilty High Low 20

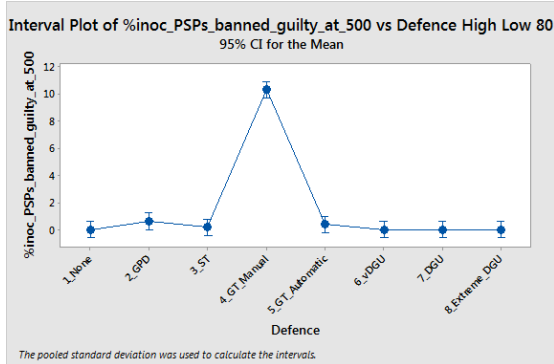


FIGURE C.70: BannedGuilty High Low 80

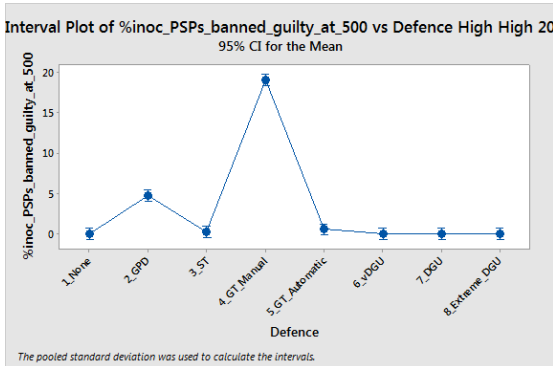


FIGURE C.71: BannedGuilty High High 20

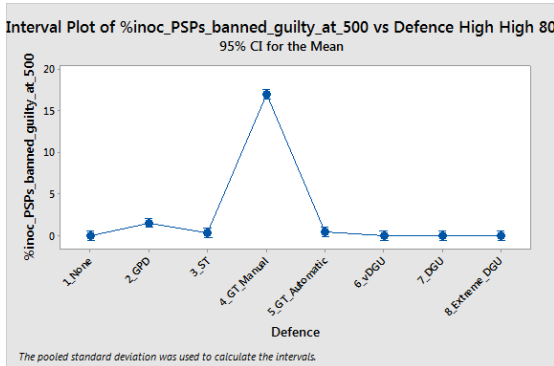


FIGURE C.72: BannedGuilty High High 80

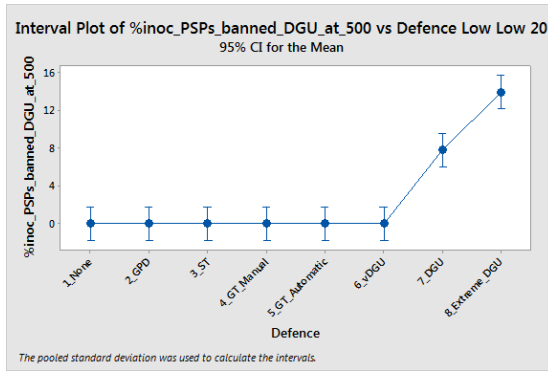


FIGURE C.73: Banned refusedInstallDGU Low Low 20

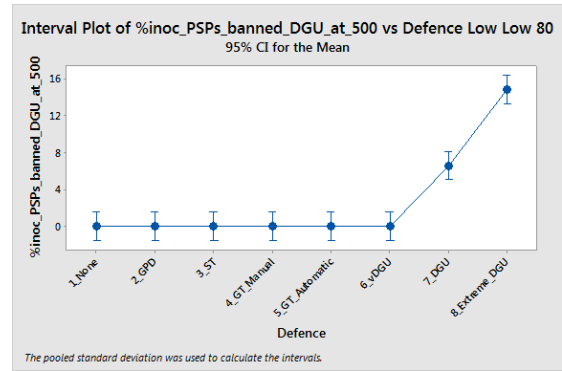


FIGURE C.74: Banned refusedInstallDGU Low Low 80

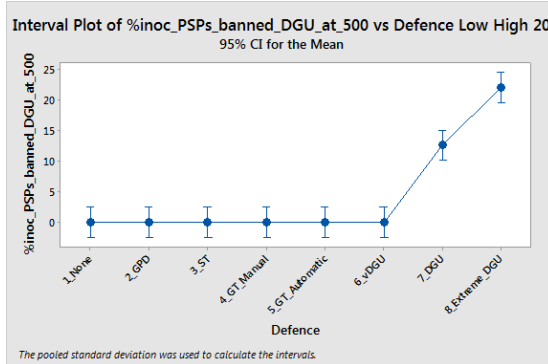


FIGURE C.75: Banned refusedInstallDGU Low High 20

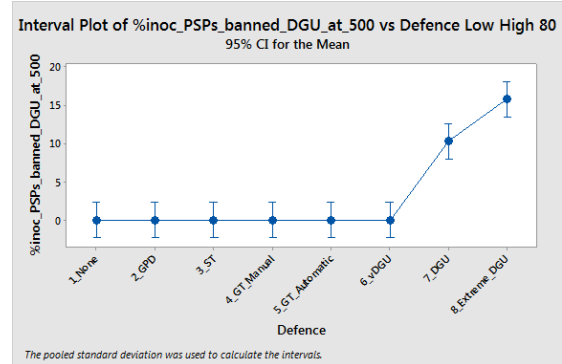


FIGURE C.76: Banned refusedInstallDGU Low High 80

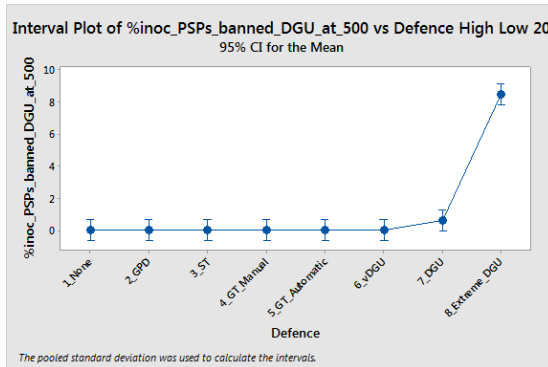


FIGURE C.77: Banned refusedInstallDGU High Low 20

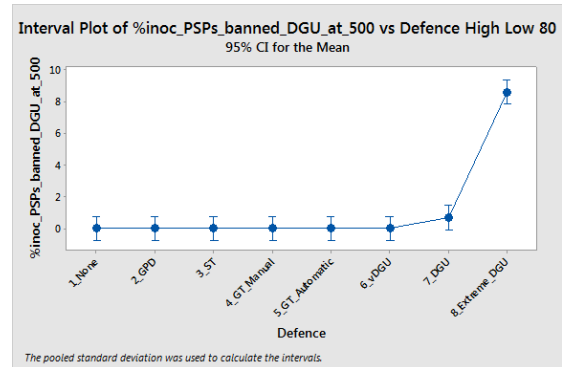


FIGURE C.78: Banned refusedInstallDGU High Low 80

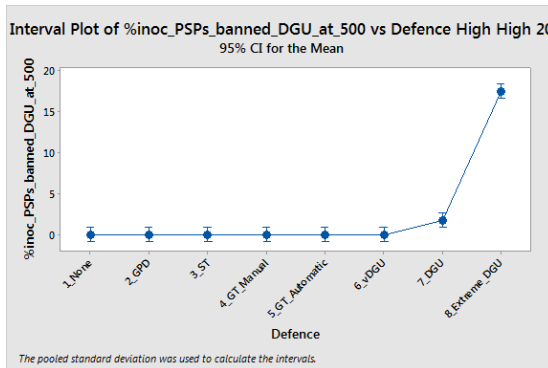


FIGURE C.79: Banned refusedInstallDGU High High 20

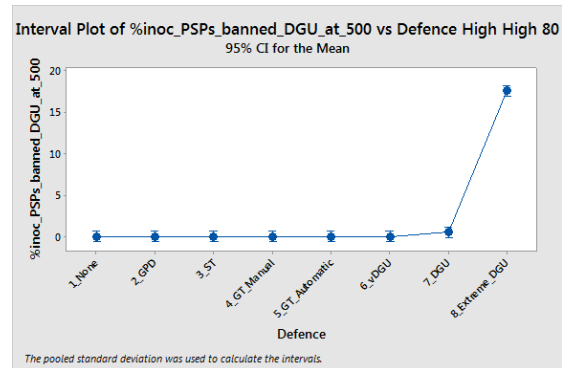


FIGURE C.80: Banned refusedInstallDGU High High 80

Bibliography

- [1] Eve Maler. A New Venn Of Access Control For The API Economy, 2012. URL http://blogs.forrester.com/eve_maler/12-03-12-a_new_venn_of_access_control_for_the_api_economy.
- [2] Jaehong Park. *Usage control: a unified framework for next generation access control*. PhD thesis, George Mason University, 2003.
- [3] Sampo Kellomäki. TAS3 Architecture. Technical report, Trusted Architecture for Securly Shared Data, 2010.
- [4] Slim Trabelsi and Akram Njeh. Policy Implementation in XACML. In Jan Camenisch, Simone Fischer-Hübner, and Kai Rannenberg, editors, *Privacy and Identity Management for Life*, pages 355–374. Springer Berlin Heidelberg, 2011.
- [5] J Camenisch, I Krontiris, A Lehmann, G Neven, C Paquin, K Rannenberg, and H Zwingelberg. Architecture for Attribute-based Credential Technologies. Technical report, Attribute-Based Credentials for Trust, 2011.
- [6] Maciej P Machulak, Eve L Maler, Domenico Catalano, and Aad van Moorsel. User-managed access to web resources. In *Proceedings of the 6th ACM workshop on Digital identity management*, DIM '10, pages 35–44, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0090-2. doi: 10.1145/1866855.1866865. URL <http://doi.acm.org/10.1145/1866855.1866865>.
- [7] OpenID-Members. Welcome to OpenID Connect, 2014.
- [8] Natsuhiko Sakimura, J Bradley, M Jones, B de Medeiros, and C Mortimore. Openid connect core 1.0. *The OpenID Foundation*, page S3, 2014.

- [9] D M Rousseau, S B Sitkin, R S Burt, C Camerer, and Others. Not so different after all: A cross-discipline view of trust. *Academy of management review*, 23(3):393–404, 1998.
- [10] H Nissenbaum. Can trust be secured online? A theoretical perspective. *Ethics & Politics*, 1999.
- [11] Stephen Paul Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, Department of Computing Science and Mathematics, University of Stirling, 1994. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.47.6243>.
- [12] D Osterwalder. Trust through evaluation and certification? *Social Science Computer Review*, 19(1):32–46, 2001.
- [13] Ping Zhang, Arjan Durrezi, and Leonard Barolli. Survey of Trust Management on Various Networks. In *Proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems*, volume 0, pages 219–226, Los Alamitos, CA, USA, 2011. IEEE Computer Society. ISBN 978-0-7695-4373-4. doi: <http://doi.ieeecomputersociety.org/10.1109/CISIS.2011.122>.
- [14] P Joshi and C.-C.J. Kuo. Security and privacy in online social networks: A survey. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, jul 2011. doi: 10.1109/ICME.2011.6012166.
- [15] S Pearson and A Benameur. Privacy, Security and Trust Issues Arising from Cloud Computing. In *Proceedings of the IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 693–702, 2010. doi: 10.1109/CloudCom.2010.66.
- [16] J Nielsen. Trust or bust: Communicating trustworthiness in web design, 1999. URL <http://www.useit.com/alertbox/990307.html>.
- [17] Robert Richardson. 2010/2011 COMPUTER CRIME AND SECURITY SURVEY. Technical report, CSI Computer Crime and Security Survey.

- [18] Marco de Vivo, Gabriela O. de Vivo, and Germinal Isern. Internet security attacks at the basic levels. *ACM SIGOPS Operating Systems Review*, 32(2):4–15, 1998. ISSN 01635980. doi: 10.1145/506133.506136. URL <http://portal.acm.org/citation.cfm?doid=506133.506136>.
- [19] M Becher, F C Freiling, J Hoffmann, T Holz, S Uellenbeck, and C Wolf. Mobile Security Catching Up? Revealing the Nuts and Bolts of the Security of Mobile Devices. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 96–111, may 2011. doi: 10.1109/SP.2011.29.
- [20] Michael A Davis. Physical and Logical Security Convergence. Technical report, InformationWeek Reports, jan 2012.
- [21] Eric S Raymond. The cathedral and the bazaar-musings on Linux and open source by an accidental revoltionary (rev. ed.), 2001.
- [22] Elias Levy. Wide open source. *SecurityFocus. com. Electronic document*. <http://www.securityfocus.com/news>, 19, 2000.
- [23] Guido Schryen. Is open source security a myth? What do vulnerability and patch data say? *Communications of the ACM (CACM)*, 54(5):130–139, 2011.
- [24] Wade Baker, Mark Goudie, Alexander Hutton, C David Hylender, Jelle Niemantsverdriet, Christopher Novak, David Ostertag, Christopher Porter, Mike Rosen, Bryan Sartin, and Peter Tippet. 2010 DATA BREACH INVESTIGATIONS REPORT: A study conducted by the Verizon Business RISK team in cooperation with the United States Secret Service. Technical report, Verizon, 2010.
- [25] Pierangela Samarati and Latanya Sweeney. Protecting Privacy when Disclosing Information: k-Anonymity and Its Enforcement through Generalization and Suppression. Technical report, 1998.

- [26] Aameek Singh and Ling Liu. TrustMe: anonymous management of trust relationships in decentralized P2P systems. In *Proceedings of the Third International Conference on Peer-to-Peer Computing*, pages 142–149, 2003. doi: 10.1109/PTP.2003.1231514.
- [27] Nicola Dragoni. A Survey on Trust-Based Web Service Provision Approaches. In *2010 Third International Conference on Dependability*, pages 83–91, jul 2010. ISBN 978-1-4244-7530-8. doi: 10.1109/DEPEND.2010.21. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5562842>.
- [28] Siani Pearson. Trusted computing: Strengths, weaknesses and further opportunities for enhancing privacy. In Peter Herrmann, Valérie Issarny, and Simon Shiu, editors, *Trust Management*, volume 3477 of *Lecture Notes in Computer Science*, pages 91–117. Springer Berlin / Heidelberg, 2005.
- [29] R Anderson. Trusted computing frequently asked questions, 2003. URL <http://www.cl.cam.ac.uk/~simrja14/tcpa-faq.html>.
- [30] Jensen J Zhao, Allen D Truell, Melody W Alexander, and Rod Davis. State e-government service and economic competitiveness: A relational analysis. *Issues in Information Systems*, VII(2), 2006.
- [31] Michael D Birnhack. The EU data protection directive: an engine of a global regime. *Computer Law & Security Review*, 24(6):508–520, 2008.
- [32] Joseph Alhadeff and Brendan Van Alsenoy. Requirements: Privacy, governance and contractual options. Technical report, Trusted Architecture for Securely Shared Data, 2009.
- [33] Felix Wu. No Easy Answers in the Fight Over iPhone Decryption. *Communications of the ACM*, 59(9), 2016.
- [34] Michael L Katz and Carl Shapiro. Antitrust in software markets. In *Competition, Innovation and the Microsoft monopoly: Antitrust in the digital marketplace*, pages 29–81. Springer, 1999.

- [35] Moshe Givon, Vijay Mahajan, and Eitan Müller. Software Piracy: Estimation of Lost Sales and the Impact on Software Diffusion. *Journal of Marketing*, 59: 29–37, 1995.
- [36] Erik Bataller. 10 Steps to Effective Data Classification. Technical report, InformationWeek Reports, 2009.
- [37] John Linn. Technology and Web user data privacy: A survey of risks and countermeasures, 2005. ISSN 15407993.
- [38] Nicola Dragoni and Fabio Massacci. Security-by-contract for web services. In *Proceedings of the 2007 ACM workshop on Secure web services, SWS '07*, pages 90–98, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-892-3. doi: 10.1145/1314418.1314433. URL <http://doi.acm.org/10.1145/1314418.1314433>.
- [39] Siani Pearson, Marco Mont, and Stephen Crane. Persistent and dynamic trust: Analysis and the related impact of trusted platforms. In Peter Herrmann, Valérie Issarny, and Simon Shiu, editors, *Trust Management*, volume 3477 of *Lecture Notes in Computer Science*, pages 407–412. Springer Berlin / Heidelberg, 2005.
- [40] R Wenning, M Schunter, L Cranor, M Marchiori, and Others. The platform for privacy preferences 1.1 (P3P1. 1) specification. *W3C Working Group Note*, 2006.
- [41] M C Mont, S Pearson, S Creese, M Goldsmith, and N Papanikolaou. EnCoRe: Towards A Conceptual Model For Privacy Policies. *PrimeLife2010*, aug 2010.
- [42] D Ferraiolo, J Cugini, and D R Kuhn. Role-based access control (RBAC): Features and motivations. In *Proceedings of 11th Annual Computer Security Application Conference*, pages 241–248. IEEE Computer Society Press, 1995.
- [43] T Moses and Et al. *extensible access control markup language (xacml) version 2.0*. Oasis Standard, 2005.
- [44] D W Chadwick and A Otenko. The PERMIS X. 509 role based privilege management infrastructure. *Future Generation Computer Systems*, 19(2):277–289, 2003.

- [45] Common-Criteria-Members. Introduction and general model. Technical report, Common Criteria for Information Technology Security Evaluation, 2009.
- [46] ISO-Members. ISO - International Organisation for Standardization, 2013. URL <http://www.iso.org/>.
- [47] IETF-Members. The Internet Engineering Task Force, 2013. URL <http://www.ietf.org/>.
- [48] W3C-Members. World Wide Web Consortium, 2013. URL <http://www.w3.org/>.
- [49] OASIS-Group. OASIS: Advancing open standards for the information society, 2013. URL <https://www.oasis-open.org/>.
- [50] Eran Hammer. OAuth 2.0 and the Road to Hell. Blog, jul 2012. URL <http://hueniverse.com/2012/07/oauth-2-0-and-the-road-to-hell/>.
- [51] Felix Schwagereit, Ansgar Scherp, and Steffen Staab. Survey on Governance of User-generated Content in Web Communities. In *ACM WebSci'11*, jun 2011.
- [52] Z Malik and A Bouguettaya. Reputation Bootstrapping for Trust Establishment among Web Services. *Internet Computing, IEEE*, 13(1):40–47, 2009. ISSN 1089-7801. doi: 10.1109/MIC.2009.17.
- [53] Sepandar D Kamvar, Mario T Schlosser, and Hector Garcia-Molina. The Eigentrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th international conference on World Wide Web, WWW '03*, pages 640–651, New York, NY, USA, 2003. ACM. ISBN 1-58113-680-3. doi: 10.1145/775152.775242. URL <http://doi.acm.org/10.1145/775152.775242>.
- [54] Li Xiong and Ling Liu. PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):843–857, jul 2004. ISSN 1041-4347. doi: 10.1109/TKDE.2004.1318566.

- [55] B C Neuman and T Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications Magazine*, 32(9):33–38, 1994.
- [56] D Hardt. *The OAuth 2.0 Authorization Framework*. IETF, jul 2012.
- [57] A Pfitzmann, K Borcea-Pfitzmann, and J Camenisch. PrimeLife. *Privacy and Identity Management for Life*, pages 5–26, 2011.
- [58] Shibboleth-Members. Shibboleth Website, 2013. URL <http://shibboleth.net/about/index.html>.
- [59] UMA-Members. User Managed Access Work Group, 2013. URL <http://kantarainitiative.org/groups/user-managed-access-work-group/>.
- [60] A C C Yao. How to generate and exchange secrets. In *the 27th Annual Symposium on Foundations of Computer Science*, pages 162–167. IEEE, 1986.
- [61] C Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 169–178. ACM, 2009.
- [62] C X Shen, H G Zhang, H M Wang, J Wang, B Zhao, F Yan, F J Yu, L Q Zhang, and M D Xu. Research on trusted computing and its development. *Science China Information Sciences*, 53(3):405–433, 2010.
- [63] Ahmad-Reza Sadeghi. Trusted Computing — Special Aspects and Challenges. In Viliam Geffert, Juhani Karhumäki, Alberto Bertoni, Bart Preneel, Pavol Návrat, and Mária Bieliková, editors, *SOFSEM 2008: Theory and Practice of Computer Science*, volume 4910 of *Lecture Notes in Computer Science*, pages 98–117. Springer Berlin / Heidelberg, 2008.
- [64] TCG-Members. *TCG Specification Architecture Overview*. Trusted Computing Group, aug 2007.
- [65] TCG-Members. Trusted Computing Group, jan 2013. URL <http://www.trustedcomputinggroup.org/>.

- [66] A Lee-Thorp. Attestation in Trusted Computing: Challenges and Potential Solutions. Technical report, 2010.
- [67] TCG-Members. More Secure Computing. Technical report, Trusted Computing Group, 2006.
- [68] Siani Pearson and Boris Balacheff. *Trusted computing platforms: TCPA technology in context*. Prentice Hall PTR, 2003.
- [69] Ernie Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *Proceedings of the 11th ACM conference on Computer and communications security, CCS '04*, pages 132–145, New York, NY, USA, 2004. ACM. ISBN 1-58113-961-6. doi: 10.1145/1030083.1030103. URL <http://doi.acm.org/10.1145/1030083.1030103>.
- [70] Common-Criteria-Members. *Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components*. Common Criteria, v 3.1 edition, 2012.
- [71] Common-Criteria-Members. Certified Products : New CC Portal, 2013. URL <http://www.commoncriteriaportal.org/products/>.
- [72] Christopher Tarnovsky. Deconstructing a 'Secure' Processor, 2010. URL https://media.blackhat.com/bh-dc-10/video/Tarnovsky_Chris/BlackHat-DC-2010-Tarnovsky-DeconstructProcessor-video.m4v.
- [73] Jaehong Park and Ravi Sandhu. Towards usage control models: beyond traditional access control. In *Proceedings of the seventh ACM symposium on Access control models and technologies, SACMAT '02*, pages 57–64, New York, NY, USA, 2002. ACM. ISBN 1-58113-496-7. doi: 10.1145/507711.507722. URL <http://doi.acm.org/10.1145/507711.507722>.
- [74] Xinwen Zhang. *Formal model and analysis of usage control*. PhD thesis, Cite-seer, 2006.
- [75] Aliaksandr Lazouski, Fabio Martinelli, and Paolo Mori. Usage control in computer security: A survey. *Computer Science Review*, 4(2):81–99, 2010.

ISSN 1574-0137. doi: 10.1016/j.cosrev.2010.02.002. URL <http://www.sciencedirect.com/science/article/pii/S1574013710000146>.

- [76] Masoom Alam, Xinwen Zhang, Mohammad Nauman, Tamleek Ali, and Jean-Pierre Seifert. Model-based behavioral attestation. In *Proceedings of the 13th ACM symposium on Access control models and technologies*, SACMAT '08, pages 175–184, New York, NY, USA, 2008. ACM.
- [77] Joseph Alhadeff and Brendan Van Alsenoy. Contractual Framework. Technical report, Trusted Architecture for Securly Shared Data, 2009.
- [78] Dennis Vandevenne, Danny De Cock, Marc Santos, Michele Bezzi, Jeroen Hoppenbrouwers, and Andreas Pashalidis. WP4 Implementation. Technical report, Trusted Architecture for Securly Shared Data, 2010.
- [79] Sampo Kellomäki. TAS3 Protocols, API, and Concrete Architecture. Technical report, Trusted Architecture for Securly Shared Data, 2009.
- [80] Ulrich Pinsdorf, Laurent Bussard, Sebastian Meissner, Jan Schallaböck, and Stuart Short. Privacy for Service Oriented Architectures. In Jan Camenisch, Simone Fischer-Hübner, and Kai Rannenberg, editors, *Privacy and Identity Management for Life*, pages 383–411. Springer Berlin Heidelberg, 2011.
- [81] M P Machulak, Ł. Moreń, and A van Moorsel. Design and implementation of user-managed access framework for web 2.0 applications. In *Proceedings of the 5th International Workshop on Middleware for Service Oriented Computing*, pages 1–6. ACM, 2010.
- [82] Facebook-Corporation. Facebook Login - Facebook Developers, 2013. URL <http://developers.facebook.com/docs/concepts/login/>.
- [83] OAuth-Community. OAuth Community Website, 2013. URL <http://www.oauth.net/>.
- [84] OpenID-Members. OpenID Website, 2013. URL <http://openid.net/>.

- [85] OpenID-Members. OpenID Connect FAQ and Q&As, 2016. URL <http://openid.net/connect/faq/>.
- [86] Wanpeng Li and Chris J Mitchell. Analysing the Security of Google’s implementation of OpenID Connect. *arXiv preprint arXiv:1508.01707*, 2015.
- [87] Vladislav Mladenov, Christian Mainka, Julian Krautwald, Florian Feldmann, and Jörg Schwenk. On the security of modern Single Sign-On Protocols: OpenID Connect 1.0. *arXiv preprint arXiv:1508.04324*, 2015.
- [88] Scott Cantor, Steven Carmody, Marlena Erdos, Keith Hazelton, Walter Hoehn, R L "Bob" Morgan, Tom Scavo, and David Wasley. Shibboleth Architecture: Protocols and Profiles. Technical report, internet2, 2005.
- [89] Vince Smith. OpenID vs. Shibboleth: power to the people. Blog, 2007.
- [90] Kerberos-Members. MIT Kerberos Consortium Website, 2007. URL <http://www.kerberos.org/>.
- [91] TCG-Members. TRUSTED COMPUTING GROUP (TCG) TIMELINE. Technical report, Trusted Computing Group, 2011.
- [92] Opus One and Igraphyx. ARCHITECT’S GUIDE: Mobile Security Using TNC Technology. Technical report, Trusted Computing Group, 2011.
- [93] CAS-Members. CAS — Jasig Community, 2009. URL <http://www.jasig.org/cas>.
- [94] Higgins-Members. Higgins Personal Data Service, 2009. URL <http://www.eclipse.org/higgins/>.
- [95] OASIS-Group. *WS-Trust 1.4*. OASIS, 2012.
- [96] John Hughes and Eve Maler. *Security Assertion Markup Language (SAML) V2.0 Technical Overview*. OASIS, 2005.
- [97] OASIS-Group. *eXtensible Access Control Markup Language (XACML) Version 2.0*. OASIS, 2005.

- [98] PrimeLife Members. PrimeLife - Privacy and Identity Management in Europe for Life, 2011. URL <http://primelife.ercim.eu/>.
- [99] Jan Camenisch, Benjamin Kellermann, Stefan Köpsell, Stefano Paraboschi, Franz-Stefan Preiss, Stefanie Pötzsch, Dave Raggett, Pierangela Samarati, and Karel Wouters. Open Source Contributions. In Jan Camenisch, Simone Fischer-Hübner, and Kai Rannenberg, editors, *Privacy and Identity Management for Life*, pages 459–477. Springer Berlin Heidelberg, 2011.
- [100] Erik Wästlund and Peter Wolkerstorfer. PET-USES. In Jan Camenisch, Simone Fischer-Hübner, and Kai Rannenberg, editors, *Privacy and Identity Management for Life*, pages 213–219. Springer Berlin Heidelberg, 2011.
- [101] PICOS-Members. PICOS - Privacy and Identity Management for Community Services, 2011. URL <http://www.picos-project.eu/>.
- [102] SSEDIC-Members. SSEDIC: Scoping the single European Digital Identity Community, 2013. URL <http://www.eid-ssedic.eu/>.
- [103] ICT-Endorse Members. Endorse: Legal Technical Framework for Privacy Preserving Data Management, 2012. URL <http://ict-endorse.eu/>.
- [104] CoSign-Members. CoSign: Web Single Sign On, 2012. URL <http://weblogin.org/>.
- [105] WebAuth-Members. Stanford WebAuth, 2013. URL <http://webauth.stanford.edu/>.
- [106] Facebook-Corporation. Facebook Developers Platform, 2013. URL <https://developers.facebook.com/>.
- [107] Flickr-Corporation. Flickr Authentication API, 2013. URL <http://www.flickr.com/services/api/auth.spec.html>.
- [108] Google-Corporation. Google Accounts Authentication and Authorization, 2013. URL <https://developers.google.com/accounts/>.

- [109] OpenID-Members. What is OpenID?, 2013. URL <http://openid.net/get-an-openid/what-is-openid/>.
- [110] Microsoft-Corporations. U-Prove, 2013. URL <http://research.microsoft.com/en-us/projects/u-prove/>.
- [111] OpenAM-Members. OpenAM Project, 2013. URL <http://openam.forgerock.org/>.
- [112] FICAM-Members. *Federal Identity, Credential, and Access Management (FICAM) Roadmap and Implementation Guidance*. ICAM, 2011.
- [113] FutureID-Members. Future ID: Shapping the Future of Electronic Identity, 2013. URL <http://www.futureid.eu/>.
- [114] Leonard Richardson and Sam Ruby. *RESTful web services*. ” O’Reilly Media, Inc.”, 2008.
- [115] Seth Ladd, Darren Davison, Steven Devijver, Colin Yates, Rob Harrop, and Keith Donald. *Expert Spring MVC and Web Flow*, volume 1. Springer, 2006.
- [116] Deepak Alur, Dan Malks, and John Crupi. *Core J2EE Patterns: Best Practices and Design Strategies*. Prentice Hall Press, Upper Saddle River, NJ, USA, 2nd edition, 2013. ISBN 0133807460, 9780133807462.
- [117] Mike Keith and Merrick Schincariol. Introduction. In *Pro JPA 2*, pages 1–14. Springer, 2013.
- [118] MIT Consortium for Kerberos Trust and Internet. MITREid Connect. URL <http://kit.mit.edu/projects/mitreid-connect>.
- [119] Justin Richer. Using OpenID Connect, 2013. URL <http://kit.mit.edu/sites/default/files/documents/OpenID-Connect-MIT-Tech-Talk-webpdf.pdf>.
- [120] Francisco Rocha, Thomas Gross, and Aad Van Moorsel. Defense-in-depth against malicious insiders in the cloud. In *Cloud Engineering (IC2E), 2013 IEEE International Conference on*, pages 88–97. IEEE, 2013.

- [121] M E J Newman. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics*, 46(5):323–351, 2005. ISSN 0010-7514. doi: 10.1080/00107510500052444. URL <http://www.tandfonline.com/doi/abs/10.1080/00107510500052444>.
- [122] Márk Jelasity, Alberto Montresor, Gian Paolo Jesi, and Spyros Voulgaris. The PeerSim project website. URL <http://peersim.sourceforge.net/>.
- [123] Gian Paolo Jesi. PeerSim HOWTO: Build a new protocol for the PeerSim 1.0 simulator. URL <http://peersim.sourceforge.net/tutorial1/tutorial1.html>.
- [124] Fred D. Davis. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly*, 13(3):319–340, 1989. ISSN 02767783. doi: 10.2307/249008. URL <http://www.jstor.org/stable/10.2307/249008>.
- [125] Juan Carlos Roca, Juan José García, and Juan José De La Vega. The importance of perceived trust, security and privacy in online trading systems. *Information Management & Computer Security*, 17(2):96–113, 2009. ISSN 0968-5227. doi: 10.1108/09685220910963983.
- [126] Patrick McCole, Elaine Ramsey, and John Williams. Trust considerations on attitudes towards online purchasing: The moderating effect of privacy and security concerns. *Journal of Business Research*, 63(9-10):1018–1024, 2010. ISSN 01482963. doi: 10.1016/j.jbusres.2009.02.025.
- [127] Douglas C Montgomery. *Design and Analysis of Experiments*, volume 2. Wiley & Sons, 2012. ISBN 0471316490. doi: 10.1198/tech.2006.s372. URL [http://cataleg.uab.cat/record=b1764873\\$\sim\\$S1*cat](http://cataleg.uab.cat/record=b1764873\simS1*cat).
- [128] R V Hogg and E A Tanis. *Probability and Statistical Inference*. Prentice Hall, 2006. ISBN 9780131293823. URL <https://books.google.com.kw/books?id=MK11AAAACAAJ>.

- [129] Minitab Inc. Designing an Experiment, 2016. URL <http://support.minitab.com/en-us/minitab/17/getting-started/designing-an-experiment/#use-the-stored-model-for-additional-analyses>.
- [130] Minitab Inc. What is a main effects plot?, 2016. URL <http://support.minitab.com/en-us/minitab/17/topic-library/modeling-statistics/anova/basics/what-is-a-main-effects-plot/>.
- [131] Minitab Inc. What is an interaction?, 2016. URL <http://support.minitab.com/en-us/minitab/17/topic-library/modeling-statistics/anova/anova-models/what-is-an-interaction/>.
- [132] Minitab Inc. What is response optimization?, 2016. URL <http://support.minitab.com/en-us/minitab/17/topic-library/modeling-statistics/using-fitted-models/response-optimization/what-is-response-optimization/>.
- [133] Suliman A Alsuhibany, Ahmad Alonaizi, Charles Morisset, Chris Smith, and Aad van Moorsel. Experimental Investigation in the Impact on Security of the Release Order of Defensive Algorithms. *Security Engineering and Intelligence Informatics*, page 321, 2013.
- [134] Guy Zyskind, Oz Nathan, and Others. Decentralizing privacy: Using blockchain to protect personal data. In *Security and Privacy Workshops (SPW), 2015 IEEE*, pages 180–184. IEEE, 2015.