Quantitative Analysis of Distributed Systems

Wen Zeng

School of Computing Science

University of Newcastle

A thesis submitted for the degree of

Doctor of Philosophy

September 2014

Acknowledgements

This work would not have been possible without intensive collaboration with many people. Firstly and foremost, I would like to express my gratitude to my supervisor Prof. Maciej Koutny, who provided me lots of support, guidance and freedom during my study. His insights and suggestions have been invaluable to my work. Next, I would like to thank my former supervisor Prof. Aad van Moorsel, for giving me the opportunity to work with Trust Economics project and for all the support he has given me over the years. I am also very grateful to Dr. Chunyan Mu and Dr. Simon Parkin for their advice throughout my study. I have greatly benefited from the close collaboration with them.

I would also like to thank my examiners, Prof. Hanna Klaudel and Dr. Andrey Mokhov, who provided encouraging and constructive feedback. It is not an easy task, reviewing a thesis, and I am grateful for their thoughtful and detailed comments.

Many thanks to other scientists, whose assistance, advice and work have helped to make this thesis possible.

I have made some great friends in the School of Computing Science. Their help and friendship made my time at the school a happy and a fruitful one. I would like to take this opportunity to thank all the people in School of Computing Science for their kindness and help.

Finally, I would like to give the credits to my parents for their love and support. Always they are close beside me, and without their support this would not have been possible.

Abstract

Computing Science addresses the security of real-life systems by using various security-oriented technologies (e.g., access control solutions and resource allocation strategies). These security technologies significantly increase the operational costs of the organizations in which systems are deployed, due to the highly dynamic, mobile and resource-constrained environments. As a result, the problem of designing user-friendly, secure and high efficiency information systems in such complex environment has become a major challenge for the developers.

In this thesis, firstly, new formal models are proposed to analyse the secure information flow in cloud computing systems. Then, the opacity of workflows in cloud computing systems is investigated, a threat model is built for cloud computing systems, and the information leakage in such system is analysed. This study can help cloud service providers and cloud subscribers to analyse the risks they take with the security of their assets and to make security related decision.

Secondly, a procedure is established to quantitatively evaluate the costs and benefits of implementing information security technologies. In this study, a formal system model for data resources in a dynamic environment is proposed, which focuses on the location of different classes of data resources as well as the users. Using such a model, the concurrent and probabilistic behaviour of the system can be analysed. Furthermore, efficient solutions are provided for the implementation of information security system based on queueing theory and stochastic Petri nets. This part of research can help information security officers to make well judged information security investment decisions.

Contents

Chapter 1

Introduction

1.1 Background and Motivation

Over recent decades, most business organizations have come to depend on information technology for administrative, service and marketing needs. Meanwhile, security and privacy have been a major concern when organizations design computer network and systems. Much research on the information security has been, traditionally, focused on the technology and products themselves. However, the impact of these security technologies on business process and human behaviour on the effectiveness of system have not been documented to date, although it has been identified that the behaviour of users has a significant impact on information security, and attacks associated with inappropriate behaviour of users are on the rise and are posing a great threat [1; 2]. Therefore, organizations have to define sets of security polices and use security technologies to enhance the security of the computer network and systems in the organization.

Extensive investment in security policies and information security technologies can lead to highly complex systems. A majority of today's security infrastructure is static. This enforces security policies defined in advance within an IT or business infrastructure, be relatively unchanged. This is no longer sufficient in an environment that is highly dynamic. Therefore, an important development in modern day security systems is recognition that means the system must achieve the optimal performance and the trade-off between the security and cost can be

balanced in the complex and dynamic environment.

Federated cloud systems increase the reliability and reduce the cost of computational support to an organization. The resulting combination of secure private clouds and less secure public clouds impacts on the security requirements of the system. Therefore, applications need to be located with different clouds, which strongly affect the information flow security of the entire system.

Information security technologies protect the confidentiality of data resources in the organization by providing access control. However, these security technologies might potentially increase the operational costs of the organizations. The mobility of the users, the different type of data resources, in conjunction with potential threats in different locations, make it difficult to analyse the risk of data exposed in different environments, and the efficiency of information security technologies.

1.2 Aims

The goal of this Thesis is to quantitatively analyse the distributed system. Firstly, we will analyse the secure information flow in the cloud computing systems, and quantitatively analyse the opacity in such systems. Secondly, we will quantitatively analyse the implementations of information security technologies, which include the security of data resources, productivity loss in the organization, and the cost of information help desks.

1.3 Contributions

The main contributions of the Thesis are summarised below:

- A lattice is proposed for federated cloud systems, and a flow-sensitive security model (*fssm*) is introduced to preserve the security properties and prevent unauthorised information flow in a federated cloud system. The *fssm* can be used to control information flow in service-oriented distributed computing systems. Moreover, the opacity and observations of the *fssm* is

analysed, and then a threat model for federated cloud computing is proposed based on the observed actions of the system. Probabilistic models are built to quantify a system's observable behaviours under any given security preserving mechanisms. Observational equivalence, entropy, channel capacity, Kullback-Leibler divergence and Jensen-Shannon divergence are used to quantify the opacity in the could computing systems. This study can be used to help cloud providers to make security related decision when they design the system.

- A procedure (QEP) is established to quantitatively evaluate the benefits and costs of implementing information security technologies in order to assist security investors to make sensible security investment decisions. Stochastic Petri nets models are used to simulate business processes, and quantify the benefits and costs of implementing information security technologies. Enterprise Digital Rights Management (EDRM) technology is used as case study, security metrics are defined as a standard to measure the impact of EDRM products. Staff productivity reduction is measured by non-productive time (NPT) and translated into monetary terms. NPT incurred by implementing EDRM product is quantified by firing delay of Petri nets.

- A system model for data resources in the organization is proposed based on the location of different classes of data resources as well as users. In this model, the status of the data resources are divided into different classes to analyse based on the security policy in the organization. Transition systems are built to track the data resources and users in different locations. Using such a model, the concurrent and probabilistic behaviours of the system can be analysed. The modelling helps us move from the debate from discussion of claims made by the information security technology vendors to a better understanding of the effectiveness of information security technology and security policies.

- An approximate solution for implementing information security technologies is analysed and evaluated based on queueing theory, and a non-productive time (NPT) function for implementing those technologies is also analysed

and evaluated. Stochastic Petri nets models are built to simulate the business processes. Multiple administrators in the information help desk is also analysed, and a cost function is built to analyse the number of administrators in the system. Moreover, sensitivity analysis is applied to determine which of the parameters exerts the most influence on the NPT of the organization, and help organization to develop a cost-effective strategy for mitigating the negative impact of implementing information security technologies. This study can help information security manager to know the number of users a given system support, and the service capacity the providers must provide at the system to satisfy a given number of users. This study also can help the organization take actions toward reducing the cost of the information security technologies.

1.4 Outline

The thesis is organized as flows:

Chapter 1 is introduction.

Chapter 2 provides the basic notions would be used in the thesis.

Chapter 3 presents a system model of information flow in federated cloud computing system and opacity of the system.

Chapter 4 presents the quantitative analysis methodology for information security technology.

Chapter 5 presents a system model for data resources in the dynamic environments.

Chapter 6 presents the efficient solutions for implementing information security technologies.

Chapter 7 is the related work.

Chapter 8 is the conclusion.

1.5 List of Publications

Below is a list of publications contributing to the Thesis and the Chapters where they are relevant:

Journals:

1. Zeng, W., Koutny, M. and van Moorsel, A. *Efficient solutions of implementing information security technologies.* (under preparation) (in Chapter 6)

2. Zeng, W., Koutny, M. and Mu, C. *Quantitative analysis of opacity in cloud computing systems.* Submitting to Journal of Computer Security (JCS) (in Chapter 3)

Conferences/Workshops:

1. Zeng, W., Koutny, M. and Watson, P. *Verifying secure information flow in federated clouds.* In 6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), December 2014. (in Chapter 3)

2. Zeng, W., Koutny, M. and van Moorsel, A. *Performance modelling and evaluation of enterprise information security technologies.* In 14th IEEE International Conference on Computer and Information Technology (CIT), September 2014. (in Chapter 6)

3. Zeng, W., Mu, C., Koutny, M. and Watson, P. *A flow sensitive security model for cloud computing systems.* In Engineering Dependable Systems of Systems (EDSoS), May 2014. (in Chapter 3)

4. Zeng, W. and Koutny, M. *Data resources in dynamic environments.* In 8th IEEE International Symposium on Theoretical Aspects of Software Engineering (TASE), September 2014. (in Chapter 5)

5. Zeng, W., Liu, K. and Koutny, M. *Cost-benefit analysis of Digital Rights Management products using stochastic models.* In 46th IEEE Annual Simulation Symposium (ANSS), April 2013. (in Chapter 4)

6. Zeng, W. and Liu, K. *Sensitivity analysis of loss of corporate efficiency and productivity associated with enterprise DRM Technology.* In 7th IEEE International Conference on Availability, Reliability and Security (ARES), August 2012. (in Chapter 6)

7. Mu, C. and Zeng, W. *Security measurement in service-based computing systems.* In 5th IEEE International Conference on Service-Oriented Computing and Applications (SOCA), December 2012. (in Chapter 3)

8. Zeng, W. and van Moorsel, A. *Quantitative evaluation of enterprise DRM technology.* In Electronic Notes in Theoretical Computer Science, Volume 275, Pages 159-174 27 September 2011. (in Chapter 6)

Chapter 2

Preliminary Material

In this part, the definition of the basic concepts and key notions are presented to follow the technical content of this thesis.

2.1 Information Theory

2.1.1 Probability Theory

The basic definition and concepts of probability theory are reviewed in this part. Probability theory [3] is concerned with the analysis of random experiences such as tossing a coin or rolling a die. A random experiment is an experiment, which can be repeated numerous times under the same conditions. The outcome of each individual experiment must be independent, identically distributed and can not be predicted with certainty. The relative frequency of an event for any random experiment is called the probability of the event. Consider the experiment of tossing a coin as an example, the experiment can yield two possible outcomes: heads and tails, both with probability of one half in any long series of trials. Probability theory normally treats discrete probability and continuous distributions separately. This thesis only considers discrete probability distributions. The definition of a discrete probability distribution is as follows:

Definition 1. Discrete probability distribution *Let D be a finite set. A discrete probability distribution on D is a function f: $D \rightarrow$ [0, 1], such that* $\sum_{d \in D} f(d) = 1$.

2.1.2 Shannon's Measure of Entropy

In order to measure the information flow, the system is treated as a communication channel. Information theory introduced the definition of entropy ($\mathcal{H}$), to measure the average uncertainty in random variables. Shannon's measures were based on a logarithmic measure of the unexpectedness in a probabilistic event. The unexpectedness of an event, which occurred with some non-zero probability p was $\log_2 \frac{1}{p}$. Therefore, the total information carried by a set of n events was computed as the weighted sum of their unexpectedness:

$$\mathcal{H} = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} \tag{2.1}$$

where, $p_i \log_2 \frac{1}{p_i} = 0$ if $p_i = 0$. This quantity is called the entropy of the set of events.

Definition 2. Discrete random variable *A discrete random variable X is a surjective function, which maps events to values of a countable set, with each value in the range having probability greater than zero, i.e. $X : D \to \mathcal{R}(D)$, where D is a finite set with a specified probability distribution, and $\mathcal{R}$ is the finite range of X.*

Shannon's information theory [4] can be used to quantify the amount of information a program may leak and the way in which this depends on the distribution of inputs.

Definition 3. Shannon entropy *X is a random variable, x ranges over the set of values which X may take, and $p(x)$ denotes the probability that X takes the value x. The entropy of a discrete random variable X is denoted by $\mathcal{H}(X)$ and is defined, in accordance:*

$$\mathcal{H}(X) = \sum_{x \in X} p(x) \log \frac{1}{p(x)} \tag{2.2}$$

The entropy measures the average information content of the set of events. In the extreme case of an event with probability 1, the entropy will be 0. On the other hand, if the distribution of the events is uniform, the entropy is maximal.

The definition of the entropy of a single random variable can be extended to a pair of random variables, which is then called joint entropy.

Definition 4. Joint entropy *The joint entropy $\mathcal{H}(X,Y)$ of a pair of discrete random variables (X, Y) with a joint distribution p(x,y) is defied as:*

$$\mathcal{H}(X,Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\frac{1}{\log_2 p(x,y)} \tag{2.3}$$

Given two random variables X and Y, the conditional entropy captures dependencies between random variables, when knowledge of one may change the information about the another.

Definition 5. Conditional entropy *The conditional entropy $\mathcal{H}(X|Y)$ measures the uncertainty about X, given the knowledge of $Y = y$. It is defined as:*

$$\mathcal{H}(X|Y) = \sum_{y} p(y)\mathcal{H}(X|Y=y) = \mathcal{H}(X,Y) - \mathcal{H}(Y) \tag{2.4}$$

If $\mathcal{H}(X|Y) = 0$, there is no uncertainty on X knowing Y; and if X and Y are independent, then $\mathcal{H}(X|Y) = \mathcal{H}(X)$, i.e., knowledge of Y does not change the uncertainty on X.

The concept of mutual information is a measure of the amount of information that one random variable contains about another. It implies the reduction in the uncertainty of one random variable due to the knowledge of the other.

Definition 6. Mutual information *Let p(x, y) denote the joint distribution of $x \in X$ and $y \in Y$. The notion of mutual information between X and Y, $\mathcal{I}(X;Y)$ is given by:*

$$\begin{aligned}
\mathcal{I}(X;Y) &= \sum_{x}\sum_{y} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} && (2.5)\\
&= H(X) - H(X|Y) && (2.6)\\
&= H(Y) - H(Y|X) && (2.7)
\end{aligned}$$

If X and Y are independent, then knowing X does not give any information about Y and vice versa, so their mutual information is zero. At the other extreme,

if X and Y are identical then all information conveyed by X is shared with Y, knowing X determines the value of Y and vice versa. As a result, in the case of identity the mutual information is the same as the uncertainty contained in X (or Y) alone, namely the entropy of X (or Y: clearly if X and Y are identical they have equal entropy).

2.2 The Information Lattices

2.2.1 Information Flow Model

An *information flow model* FM [5] as follows:

$$FM = (N, P, S, \oplus, \rightarrow)$$

where $N = \{a, b, \dots\}$ is a set of logical storage *object* or information receptacles. $P = \{p, q, \dots\}$ is a set of processes. Processes are the active agents responsible for all information flow. $S = \{A, B, \dots\}$ is a set of security classes corresponding to disjoint classes of information. Each object a is bound to a security class. Each process p may also be bound to a security class.

The class-combining operator " $\oplus$ " is an associative and commutative binary operator that specifies, for any pair of operand classes, the class in which the result of any binary function on values from the operand classes belongs. The set of security classes is closed under " $\oplus$ ".

A flow relation " $\rightarrow$ " is defined on pairs of security class. For classes A and B, we write $A \rightarrow B$ if and only if information in class A is permitted to flow into class B. The security requirements of the model is: a flow model FM is *secure* if and only if execution of a sequence of operations cannot give rise to a flow that violates the relation " $\rightarrow$ ".

2.2.2 Information Lattice

A lattice for security concerns $\mathcal{L} = (L, \leq)$ consists of a set L and a partial order relation $\leq$ such that, for all $l, l' \in L$, there exists a least upper bound $l \oplus l' \in L$ and a greatest lower bound $l \otimes l' \in L$ (The lattice operation $\oplus$ and $\otimes$ are useful

for computing the security level of information that is produced by combining information at different security levels). The lattice is complete if each subset L' of L has both a least upper bound $\coprod L'$ and a greatest lower bound $\prod L'$ [5; 6]. When two elements l_1 and l_2 are ordered $l_1 \leq l_2$, then the usage of information at level l_2 is at least as restrictive as usage of information at level l_1.

Example 1. Figure 2.1 shows a structure satisfying the lattice property

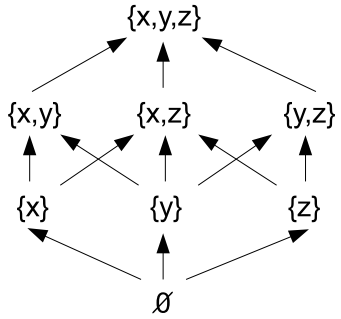


Figure 2.1: A non-linear ordered lattice

which is derived from a non-linear ordering on the set of all subsets of a given finite set $X = \{x, y, z\}$. In this figure, $S = powerset(X)$, $A \rightarrow B$ iff $A \subseteq B$, $A \oplus B = A \cup B$, $A \otimes B = A \cap B$, greatest lower bound $\prod L' = \emptyset$, and least upper bound $\coprod L' = X$. This structure is suitable when X is regarded as a set of properties and classes as combinations of properties from X. Information in an object a is not permitted to flow into an object b unless b has at least the properties of a. □

2.3 Petri Nets

Petri net is a graphical and mathematical modelling tool for the formal description of systems whose dynamics are characterized by concurrency, synchronization, mutual exclusion and conflict [7]. Petri nets have been widely used for structural modelling of work-flows and have been applied to a wide range of qualitative and quantitative analyses [7; 8; 9; 10; 11].

Definition 7. The definition of a classic Petri nets:

A Petri net is a 5-tuple $N = (P, T, F, W, M_o)$*, where:*

$P = \{p_1, p_2, \ldots, p_n\}$, *$P$ is the set of places, p_n is the name of each place;*

$T = \{t_1, t_2, \ldots, t_n\}$, *$T$ is the set transitions, t_n is the name of each transition;*

$F \subseteq (P \times T) \cup (T \times P)$ *is a set of arcs;*

$W : F \to \{1, 2, 3, \ldots\}$ *is a weight function;*

$M_0 : P \to \{0, 1, 2, 3, \ldots\}$ *is the initial marking;*

$P \cap T = \emptyset$ *and* $P \cup T \neq \emptyset$.

A Petri net consist of *place, transition* and *arcs* that connect them. In diagrams, places are drawn as circles, and transitions as rectangles. If $W(x, y) \geq 1$ for some $(x, y) \in (T \times P) \cup (P \times T)$, then (x, y) is an *arc* leading from x to y. The *pre-* and *post-multiset* of a transition $t \in T$ are multisets of places, $PRE_N(t)$ and $POST_N(t)$, respectively given by:

$$PRE_N(t)(p) \stackrel{df}{=} W(p, t) \quad and \quad POST_N(t)(p) \stackrel{df}{=} W(t, p)$$

for all $p \in P$. A *marking* of a net N is a multiset of places. Given a marking M of N and a place $p \in P$, we say that p is marked, if $M(p) \geq 1$ and that $M(p)$ is the number of tokens in p. In diagrams, M is represented by drawing in each place p exactly $M(p)$ tokens.

Transitions represent actions which may occur at a given marking and then lead to a new marking. A transition t is enabled at a marking M, if $PRE_N(t) \leq M$. In order for t to be enabled at M, for each place p, the number of tokens in p under M should at least be equal to the total number of tokens that are needed as an input to t, respecting the weights of the input arcs. If t is enabled at M, then it can be executed leading to the marking:

$$M' \stackrel{df}{=} M - PRE_N(t) + POST_N(t)$$

Hence the execution of t consumes from each place p exactly $W(p, t)$ tokens and produces in each place p exactly $W(t, p)$ tokens. If the execution of t leads from

M to M', we write $M[t\rangle M'$. We call a marking M' reachable from marking M if $M = M'$, or if there exist transitions $t_1, \ldots, t_n$ and $M_1, \ldots, M_{n-1} (n \geq 1)$ such that:

$$M[t_1\rangle M_1 \ldots M_{i-1}[t_i\rangle M_i \ldots M_{n-1}[t_n\rangle M'$$

Example 2. Figure 2.2 shows a classic Petri net model, the circles p_1, p_2, p_3

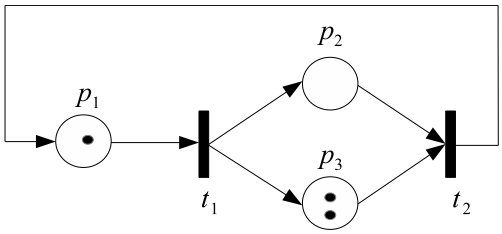


Figure 2.2: A Petri net model

represent places, p_1 contains one token and p_3 contains two tokens, vertical lines t_1, t_2 represent transitions. □

2.3.1 Coloured Petri Nets

In standard Petri nets, there is only one kind of token and the state of a place is described by an integer. In coloured Petri nets (CPNs), each token can carry complex information or data [12]. To define coloured Petri nets formalism by Jensen [12], we recall some syntax for writing the expression:

- S_{ms} denotes the multi-set over a set S. A multi-set is the same as a set, except that there may be multiple appearance of the same element.
- $Type(v)$ denotes the type of a variable v.
- $Var(expr)$ denotes the set of variables in expression expr.
- A binding of a set of variables, V associating with each variable $v \in V$ an element $b(v) \in Type(v)$.
- The value obtained by evaluating an expression, $expr$, in a binding, b denoted by $expr\langle b\rangle$. $Var(expr)$ is required to be a subset of the variables

of b, and the evaluation is performed by substituting for each variable $v \in Var(expr)$ the value $b(v) \in Type(v)$ determined by the binding.

Definition 8. CPN: *A tuple* $CPN = (\sum, P, T, A, C, G, E, I)$ *be a Coloured Petri Nets:*

- $\sum$ *is a finite set of non-empty sets, called colour sets.*
- P *is a finite set of places.*
- T *is a finite set of transitions.*
- A *is a finite set of arcs:* $P \cap T = P \cap A = T \cap A = \emptyset$.
- C *is a colour function:* $P \rightarrow \sum$.
- G *is a guard function:*

  $\forall t \in T : [Type(G(t)) = B \wedge Type(Var(G(t))) \subseteq \sum],\ B = \{true, false\}$.
- E *is an expression function:*

  $\forall a \in A : [Type(E(a)) = C(p(a))_{MS} \wedge Type(Var(E(t))) \subseteq \sum]$.
- I *is an initialization function:*

  $\forall p \in P : [Type(I(P)) = C(p)_{MS}]$.

The colour function C maps each place, p, to a colour set $C(p)$. The guard function G maps each transition, t, to an expression of type boolean. The arc expression function E maps each arc, a, into an expression which must be of type $C(p(a))_{MS}$. The initialization function I maps each place, p, into a closed expression which must be of type $C(p)_{MS}$.

Definition 9. Workflow based on Coloured Petri nets (WFCP-nets): *A tuple* $CPN = (\sum, P, T, A, C, G, E, I)$ *is a WFCP-net if and only if:*

- IN *and* OUT *are subsets of* P*;* IN *which has one element is a set of workflow start places,* OUT *may have one or many elements is a set of terminal places formal description:* $IN, OUT \in P : [|IN| = 1; |OUT| \geq 1]$ *and* $\forall i \in IN, {}^\bullet i = \emptyset; \forall o \in OUT, o^\bullet = \emptyset$.

- $\forall x \in P \cup T \wedge x \notin IN \wedge x \notin OUT$, *x is on the path from* $i \in IN$ *to* $o \in OUT$.
- *Tp is a category of transition T:* $\forall t \in T : Type(Tp(t)) \in TT$;
  $TT = \{Auto, User, Event, Time, Dumb\}$.
- *The initialization function:* $I = \left\{ \begin{array}{ll} C(i)_{MS}, & p = i \\ \emptyset, & p \neq i \end{array} \right\}$.

2.3.1.1 The Execution of CPN

The behaviour properties of the work-flow net reflect the run time aspects of work-flow. The enactment of the nets depends on the interpretation of arc expression and guard expression at runtime. The colour may be changed by application agents or user iteration activities during runtime.

Definition 10. Transition enabled: *An transition t is enabled in marking M if the following property is satisfied:*

$$\sum_{\forall p \in {}^\bullet t} E(p,t)\langle b \rangle \leq M(p)$$

$$G(t)\langle b \rangle = true$$

The expression evaluation $E(p,t)\langle b \rangle$ yields the multi-set of $C(p)$, if transition t connects from p to t. Transition t should satisfy logic conditions.

Definition 11. Transition execution: *As soon as enabled transition is completed, it triggers the enabling of other transitions. Formally, it changes the marking (state)* M_1 *to marking (state)* M_2*, defined by:*
$\forall p \in P : M_2(p) = (M_1(p) - \sum E(p,t)\langle b \rangle) + \sum E(t,p)\langle b \rangle$

We can say M_2 is directly reachable from M_1 by the occurrence of the transition t in binding b, which is denoted as: $M_1[t > M_2$.

Definition 12. Flow terminal: *An instance of flow terminates if the data reaches all of the terminal places, formally:* $\forall o \in OUT, C(o) \neq \emptyset$.

2.3.2 Stochastic Petri Nets

For Petri nets, a firing delay can be associated with each transition; this firing delay specifies the time that the transition has to be enabled, before it can actually fire. If the delay is a random distribution function, the Petri net is called a Stochastic Petri net.

Stochastic Activity Networks (SANs) are stochastic extensions to Petri nets. SANs consist of four primitive objects: *places, transitions, input gates* and *output gates*. A place represents each state of the modelled system. Transitions represent actions of the modeled system that take some specified amount of time to complete. They are of two types: timed and instantaneous. Timed transitions have durations that impact the performance of the modelled system. Instantaneous transitions represent transitions that complete immediately when enabled in the system. Input gates are used to control the enabling of transitions and define the marking changes that will occur when a transition completes; and output gates are used to define the marking changes that will occur when transitions complete.

Example 3. In figure 2.3, the circles p_1, p_2 represent places, and each

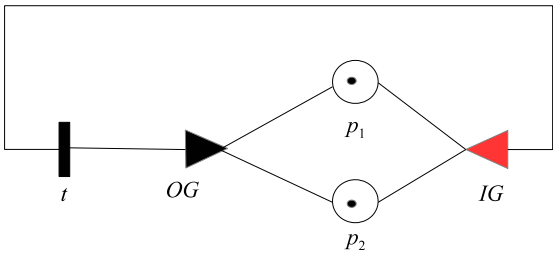


Figure 2.3: A stochastic Petri nets model

place contains a token, thick vertical line t represents timed transition, and red triangle IG represents input gate (the triangle with its tip pointing to the left), and dark triangle OG represents output gate (the triangle with its tip pointing to the right).

If the input predicate of the input gate is: $(p_1 \rightarrow Mark() == 1)\&\&(p_2 \rightarrow Mark() == 1)$, then, the predicates return true, and t is enabled. Meanwhile, if the output function is: $p_1 \rightarrow Mark() + +$; $p_2 \rightarrow Mark() + +$, then, in p_1 and p_2 the tokens will be increased by one more. □

Reward formalism: *Reward models* are used to specify measures of system behaviour [13]. Reward model has two different reward structures: one is *rate rewards*, that is, the rate at which reward accumulates while the process is in the state during an interval of time; the other is *impulse rewards*, which is used to count the number of times a transition fires during an interval of time.

Reward structure: The functions to express transition and marking oriented reward structure of a SAN with places P and transitions A:

$\mathcal{C} : A \to \mathcal{R}$, where for $a \in A$, $\mathcal{C}_a$ is the reward obtained due to completion of transition a;

$\mathcal{R} : \mathcal{P}(P, \mathcal{N}) \to \mathcal{R}$, where for $v \in \mathcal{P}(P, \mathcal{N})$, $\mathcal{R}_{(v)}$ is the rate of reward obtained when for each $(p, n) \in v$; there are n tokens in place p.

$\mathcal{N}$ is the set of natural numbers, $\mathcal{P}(P, \mathcal{N})$ is the set of all partial functions between P and $\mathcal{N}$. *Impulse rewards* are associated with transition completion (via $\mathcal{C}$) and *rates rewards* are associated with number of taken in sets of places (via $\mathcal{R}$).

$$Y_{[t,t+l]} = \sum_{v \in \mathcal{P}(P,\mathcal{N})} \mathcal{R}_{(v)} M^{v}_{[t,t+l]} + \sum_{a \in A} \mathcal{C}_a N^{a}_{[t,t+l]}$$

In this function, reward accumulated is related to the number of times each transition completes and time spent in particular markings during an interval of time $[t, t + l]$. $M^{v}_{[t,t+l]}$ represents the total time that the SAN is in a marking such that for each $v \in \mathcal{P}(P, \mathcal{N})$, there are n tokens in p during $[t, t + l]$. $N^{a}_{[t,t+l]}$ represents the number of completion of transition a during $[t, t + l]$.

2.4 Queueing Networks

Consider a system where jobs arrive, remain for some time, and then depart. The system has been running for a long time and its behaviour no longer exhibits any trends. It continues to change state, the number of jobs in it continues to vary, but the probability of observing it in any given state is no longer a function of time. Therefore, we call the systems are in steady state [14].

Definition 13. Little's Law *Suppose we observe the queueing process for an interval of time, the system reaches steady state. The arrival rate, λ, does not*

change with time and is equal to the departure rate. The average number of jobs in the system, L, does not change. The average interval, W, between the arrival of a job and its departure does not change. The following relation holds:

$$L = \lambda W \tag{2.8}$$

A queueing network can be thought of as a connected directed graph whose nodes represent service centres. A network without external arrivals and without departures, but with a fixed number of jobs circulating forever among the nodes, is called "closed queueing network" [14].

Suppose that a network is in steady state. Let λ_i be the total average number of jobs arriving into, and departing from, node i per unit time. Some of the incoming jobs be external with rate γ_i. On the average, λ_j jobs leave node j per unit time, a fraction q_{ji} go to node i. Thus, the rate of traffic from node j to node i is $\lambda_j q_{ji}, (j = 1, 2, \ldots, N)$. Figure 2.4 shows the arrives and departures at node i.

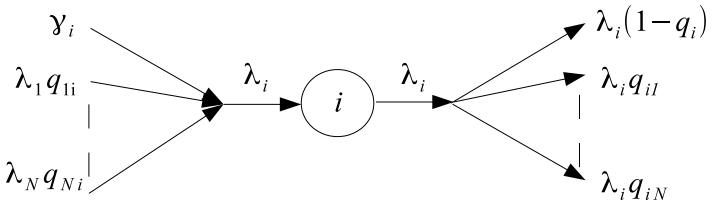


Figure 2.4: Incoming and outgoing traffic at node i [14].

Expressing the total arrival rate at each node as a sum of external and internal traffic rates, we get:

$$\lambda_i = \gamma_i + \sum_{j=1}^{N} \lambda_j q_{ji} \quad ; \quad i = 1, 2, \ldots, N. \tag{2.9}$$

These are know as the "traffic equations" of the network. The condition for stability of the network depends on the offered loads at all nodes, which in turn depend on the parameters b_i. The following must be hold:

$$\rho_i = \lambda_i b_i < 1 \quad ; \quad i = 1, 2, \ldots, N. \tag{2.10}$$

where, λ_i and b_i be the arrival rate and the average service requirement at node i.

Example 4. In figure 2.5, the external arrival process has rate γ, after

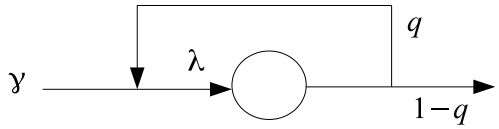


Figure 2.5: A single-node network with feedback [14].

completing service, jobs return to the queue with probability q, and leave the system with probability $1 - q$. There is one traffic equation, which determines the total arrival rate λ:

$$\lambda = \gamma + \lambda q$$

This yields $\lambda = \frac{\gamma}{1-q}$. If the average service time is b, then the condition for stability of this system is $\gamma b < 1 - q$. The traffic rate along the feedback arc is $\lambda q = \frac{\gamma q}{1-q}$. The rate of departures from the network is $\lambda(1 - q)$, which is equal to the external arrival rate γ. □

2.5 Markov Chains and Processes

A stochastic process is a mathematical model used to describe the evolution of an empirical process or system which exhibits some probabilistic behaviour. Such processes can be defined by the set of all possible states that they may reach and the set of probabilities that specify the transitions between possible states. A stochastic process is defined as follows:

Definition 14. *A stochastic process is a family of random variables* $\{X(t), t \in T\}$ *defined over the same probability space and taking values in the set* S*. The parameter* t *denotes time and it is used to index each random variable.* S *contains the values which can be obtained by the stochastic process, known as states, and therefore* S *is called the state space of the process*

If the state space is discrete then the corresponding stochastic process is said to be a discrete-state process or chain. If state space is continuous, it is called

a continuous space process. The parameter set T, i.e., the values of t, can be either discrete or continuous. in the later case the stochastic process is called a continuous-time process while in the former case it is referred to as discrete-time process or sometimes as a stochastic sequence.

2.5.1 Markov Chains

Let $X = \{X_t : t = 0, 1, \dots\}$ be a Markov chain whose state space maybe finite or infinite. The Markov property says, that given the state of X at time t, its state at time $t+1$ is independent of the states at times $0, 1, \dots, t-1$:

$$P(X_{t+1} = j | X_0, X_1, \dots, X_t) = P(X_{t+1} = j | X_t) \tag{2.11}$$

The evolution of the Markov chain is completely described by the "one-step transition probability", $q_{i,j}(t)$, that the chain will move to state j at time $t+1$, given that it is in state i at time t:

$$q_{i,j}(t) = P(X_{t+1} = j | X_t = i) \quad ; \quad i, j, t = 0, 1, \dots \tag{2.12}$$

The one-step transition probability do not depend on the time instant:

$$q_{i,j}(t) = q_{i,j} \quad ; \quad i, j, t = 0, 1, \dots \tag{2.13}$$

Markov chains which satisfy (2.13) are called to be "time-homogeneous" [14].

A Markov chain is characterized by its "transition probability matrix", Q, containing the one-step transition probability:

$$Q = \begin{pmatrix} q_{0,0} & q_{0,1} & q_{0,2} & \dots \\ q_{1,0} & q_{1,1} & q_{1,2} & \dots \\ \vdots & \vdots & \vdots & \\ q_{i,0} & q_{i,1} & q_{i,2} & \dots \\ \vdots & \vdots & \vdots & \end{pmatrix} \tag{2.14}$$

where, the indices range over the state space. All row sums of the transition

probability matrix are equal to 1:

$$\sum_{j=0}^{\infty} q_{i,j} = 1 \quad ; \quad i = 0, 1, \dots \tag{2.15}$$

Example 5. A machine can be in one of two possible states: broken or

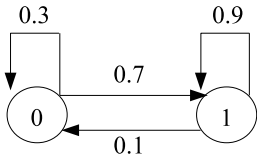


Figure 2.6: A machine with two possible states

operative (0 or 1, respectively) [14]. If it is broken at time t, it will be broken or operative at time $t+1$ with probability 0.3 and 0.7, respectively. If it is operative at time t, it will be broken or operative at time $t+1$ with probability 0.1 and 0.9, respectively ($t = 0, 1, \dots$). This behaviour satisfy the Markov property. Figure 2.6 shows the two-state Markov chain, and the transition probability matrix is:

$$Q = \begin{pmatrix} 0.3 & 0.7 \\ 0.1 & 0.9 \end{pmatrix}$$

□

2.6 Labelled Transition System

Definition 15. Sequences *The set of finite sequence over a set A is denoted by A^*, and the empty sequence is denoted by ϵ. The length n of a finite sequence $\lambda = l_1, \dots, l_n$ is denoted by $LEN(\lambda)$. The projection of λ onto a set $B \subseteq A$, denoted by $\lambda|_B$, is formed by stripping from λ all the elements that are not in B. The concatenation of sequences $\lambda_1, \dots, \lambda_n$ will be denoted by $\lambda_1 \dots \lambda_n$.*

Definition 16. Labelled transtion system *A Labelled transition system is a tuple:*

$$LTS \stackrel{df}{=} (S, L, \Delta, S_0)$$

where S is the set of states, L is the set of labels, $\Delta \subseteq S \times L \times S$ is the transition relation, and S_0 is the non-empty (finite) set of initial states. If only considering deterministic labelled transition systems, for any transitions $(s, l, s'), (s, l, s'') \in \Delta$, it is the case that $s' = s''$.

A run of the labelled transition system is a pair (s_0, λ), where $s_0 \in S_0$, and $\lambda = l_1 \dots l_n, (n \geq 0)$ is a finite sequence of labels such that there are states $s_1, \dots, s_n$ satisfying (s_{i-1}, l_i, s_i), for $i = 1, \dots, n$. The state s_n is denoted by $s_0 \oplus \lambda$, and is called *reachable* from s. $s_0 \oplus \lambda$ is well-defined since LTS is deterministic, and that $s_0 \oplus \epsilon = s_0$.

The set of all runs is denoted by $RUN(LTS)$, and the language generated by LTS is defined as:

$$\mathcal{L}(LTS) \stackrel{df}{=} \{\lambda \mid \exists s_0 \in S_0 : (s_0, \lambda) \in RUN(LTS)\}$$

A marked Petri net $PN \stackrel{df}{=} (P, T, F, W, M_0)$ has a finite set of initial markings M_0. It generates the labelled transition system:

$$LTS_{PN} \stackrel{df}{=} (M, L, \Delta, M_0)$$

where M is the set of all the markings reachable from the marking in M_0, T is the set of labels, and Δ is defined by $(M, t, M') \in \Delta$ iff $M[t\rangle M'$. The language of PN is that of LTS_{PN}.

2.7 Observability and Opacity

Opacity has been shown to be a promising technique for describing and unifying security properties. The essential ideal is that a predicate is opaque if an observer of the system will never be able to determine the truth of that predicate [15; 16].

$LTS \stackrel{df}{=} (S, L, \Delta, S_0)$ is a labelled transition system, and Obs be a set of elements called *observables*, and $Obs^\epsilon \stackrel{df}{=} Obs \cup \{\epsilon\}$.

Definition 17. Static observation function *$obs : RUN(LTS) \to Obs^*$ is a (label-based) observation function. It is called* static *if there is a map $obs' : L \to$*

Obs^{ϵ} such that for every run $(s, \lambda) = (s, l_1 \dots l_n)$ of LTS,

$$obs(s, \lambda) \stackrel{df}{=} obs'(l_1)obs'(l_2) \dots obs'(l_n)$$

obs' is allowed to return ϵ which means that one can model invisible actions. Based on the observation function, an observer might establish property *PROP* (a predicate over system runs) for some run having only access to the result of the observation function. The *PROP* is identified by its characteristic set: the set of runs for which it holds. Given an observed execution of the system, we would find out whether the fact that the underlying run belongs to PROP can be deduced by the observer.

Definition 18. The formalisation of opacity *Let obs be an observation function and PROP be a predicate PROP over RUN(LTS). Then PROP is* opaque *w.r.t. obs if, for every run $(s, \lambda) \in PROP$, there is a run $(s', \lambda') \notin PROP$, such that $obs(s, \lambda) = obs(s', \lambda')$. In other words,*

$$obs(RUN(LTS) \cap PROP) \subseteq obs(RUN(LTS \setminus PROP))$$

Moreover, *PROP* is called *final-opaque* if there is a predicate $PROP'$ such that for every run $(s, \lambda) = (s, l_1 \dots l_n)$ of *LTS*, $PROP(s, \lambda) \stackrel{df}{=} PROP'(\sigma)$, where the following hold: $\sigma \stackrel{df}{=} s \oplus (l_1 \dots l_n)$. Final-opacity models situations where the final result of a computation needs to be secret.

Proposition 1. *Let $PROP$ and $PROP'$ be two predicates over the runs of a labelled transition system such that $PROP'$ implies $PROP$. If $PROP$ is opaque w.r.t. an observation function obs then $PROP'$ is also opaque w.r.t. obs.*

2.8 Project Economics

For corporations and organizations to make sound decisions on the implementation of a project, all the associated costs and benefits over a multi-year period have to be taken into account. The concept of net present value (NPV) from economics, project management and decision making science is adopted to quantify the value of an implementation project. NPV takes the time value of money into

consideration. The basic concept is that money spent or obtained in the future will have a discounted value than money spent or obtained in the present [17]. NPV calculation is based on the principle of discounting; all the future cash flows are discounted back to the present time, which means one pound today is worth $(1+i)^t$ pound at time t in the future.

$$NPV(i,T) = -K + \sum_{t=0}^{T} \frac{R_t - C_t}{(1+i)^t}$$

In which, $-K$ is a capital costs now (K at time zero), t is the time of the cash flow, R_t is the benefits return of year t, C_t is ongoing costs which is the cost of year t, i is the discount rate. Organizations usually have its own defined discount rate, although this discount rate is in general tied to the national interest rate.

NPV is one of the most widely used decision making criteria. If the NPV of a project is positive, this project will generate positive cash flow to the organization over a certain period, therefore, is beneficiary or profitable to the organization. However, when the capital of an organization with a portfolio of many profitable projects ($NPV > 0$) is constrained, not all the project with a positive NPV can be invested. The organization usually set a value larger than 0; projects with higher NPV usually have higher priorities for investment.

Chapter 3

Information Flow Security in Cloud Computing Systems

The extent and importance of cloud computing is rapidly increasing due to the ever increasing demand for internet services and communications. Instead of building individual information technology infrastructure to host databases or software, a third party can host them in its large server clouds. However, large organizations may wish to keep sensitive information on their more restricted servers rather than in the public cloud. This has led to the introduction of federated cloud computing (FCC) in which both public and private cloud computing resources are used [18].

A federated cloud is the deployment and management of multiple could computing services with the aim of matching business needs. Data, services, and software are required to be allocated in different clouds for both security and business concerns. Although federated cloud systems (FCSs) can increase the reliability and reduce the cost of computational support to an organization, the large number of services and data on a cloud system creates security risks. As a result, it is very hard for an organization to track and control the information flow in the system. It is therefore necessary to develop a formal model describing the information flow security within an FCS, making the information and data traceable.

In this chapter, we will introduce a transition system representation to capture

the information flow in a federated cloud system, and investigate the opacity and observations of the cloud computing system. In addition, we will quantitatively analyse the opacity of such system.

3.1 Information Lattices

In this section, we introduce security lattices for the components of a cloud system as well as for sets of individual clouds.

The information lattice is represented in Section 2.2, in this part, we will assume that the origin of a federated cloud is a set $\mathcal{C}$ of single deployment clouds. Moreover, $\mathcal{S}$ will denote subjects (e.g. services, programs and processes), and O will denote objects (e.g. resources and messages). Subjects and objects will jointly be referred to as entities, and their set will be denoted by $\mathcal{E}$.

Following [6], at the centre of our approach is a complete security lattice $\mathcal{L}_{sec} = (L_{sec}, \leq_{sec})$ with the ordering $\leq_{sec}$ on security levels in L_{sec}.

We will assign a security (or confidentiality) level $l(e) \in \mathcal{L}_{sec}$ to any entity $e \in \mathcal{E}$ which will in practice be related to the degree of security of the contents of e. Moreover, each cloud $c \in \mathcal{C}$ will also be assigned a security level $l(c) \in \mathcal{L}_{sec}$. Intuitively, $l(c)$ specifies the highest allowed security level of the entities located in c.

3.2 System Model

Information flow security is concerned with the way in which secure information is allowed to flow through a computing system. Intuitively, the flow is considered *secure* if it adheres to a specified security policy. In this section, a formal model is introduced for capturing the information flow in federated cloud computing systems. The state transitions of the model can then be analyzed to verify that they satisfy conditions of a given security policy such as non-interference properties [19], Bell-Lapadula rules [20] for confidentiality considerations, Biba policies [21], and user-specified policies.

We now introduce a framework for security models of federated cloud computing systems based on guarded actions, each guard capturing security constraints

introduced in order to ensure the security properties of interest.

Throughout this section, we assume that $\mathcal{C}$ and $\mathcal{E}$ are finite non-empty sets of respectively *clouds* and *entities* (or entity names), and $\mathcal{L}_{sec}$ is a security lattice as defined above. We also assume that there is a mapping $l : \mathcal{C} \cup \mathcal{E} \to L_{sec}$ assigning security levels to individual clouds and entities.

In what follows, an entity can have different copies, and each of these copies can have a different security level and may reside in a different cloud. We further allow multiple copies of a single entity to be present in a single deployment cloud. As a result, in what follows, we will mean by a *state* any finite multiset s over the set $\mathcal{E} \times \mathcal{C} \times L_{sec}$. Thus, for example, if $s(a, c, 2) = 4$ then we know that in the current state there are 4 copies of entity a with security level 2 residing in cloud c. The elements of $\mathcal{E} \times \mathcal{C} \times L_{sec}$ will be referred to as *actual* entities. We will say that an actual entity (e, c, l) is *present* in state s if $s(e, c, l) > 0$.

Definition 19 (fssm)**.** *A* flow-sensitive security model *is a pair*

$$FSSM = (\mathcal{A}, s_{init}) \,, \tag{3.1}$$

where $\mathcal{A}$ is a finite set of actions and s_{init} is an initial state. It is assumed that each action is a triple

$$\phi = (in, out, \Sigma) \tag{3.2}$$

such that the first two components

$$in = (e_1@c_1, \ldots, e_k@c_k) \quad and \quad out = (e_{k+1}@c_{k+1}, \ldots, e_{k+m}@c_{k+m})$$

are finite tuples of entity-cloud pairs and $\Sigma \subseteq L_{sec}^{k+m}$ is a (security) guard.

Note that security guards can be provided in the form of a suitable predicate over $k + m$ variables and suitable constants (such as security levels of given clouds).[1]

Definition 20 (single action execution)**.** *An action ϕ as in (3.2) is* σ-enabled *at*

[1] One may extend the class of allowed action types to include, for example, checking of absence of certain entities.

state s *if* $\sigma = (l_1, \ldots, l_{k+m}) \in \Sigma$ *is a tuple of security levels such that*[1]

$$\phi_\sigma^{in} = \{(e_1, c_1, l_1), \ldots, (e_k, c_k, l_k)\} \leq s \ .$$

Such an action can then be σ-executed *leading to a new state*[2]

$$s' = s - \phi_\sigma^{in} + \phi_\sigma^{out} \ ,$$

where $\phi_\sigma^{out} = \{(e_{k+1}, c_{k+1}, l_{k+1}), \ldots, (e_{k+m}, c_{k+m}, l_{k+m})\}$. *We denote this by* $s \xrightarrow{\phi:\sigma}_\diamond s'$. *The step semantics describe how the individual steps of a computation take place in the system.*

The last definition captures the enabling and execution of a single action. The next definition lifts this to any group of actions executed simultaneously.

Definition 21 (multiset action execution)**.** *Let* $\Phi = \{\phi_1 : \sigma_1, \ldots, \phi_n : \sigma_n\}$ *be a multiset such that each* ϕ_i *is an action* σ_i*-enabled at state* s *and, moreover,*

$$\Phi^{in} = (\phi_1)_{\sigma_1}^{in} + \cdots + (\phi_n)_{\sigma_n}^{in} \leq s \ .$$

Then Φ *can then be* executed *leading to a new state*

$$s' = s - \Phi^{in} + \Phi^{out} \ ,$$

where $\Phi^{out} = (\phi_1)_{\sigma_1}^{out} + \cdots + (\phi_n)_{\sigma_n}^{out}$. *We denote this by* $s \xrightarrow{\Phi}_\diamond s'$. *The step semantics here describe how the overall results of the executions are obtained.*

Definition 22 (reachable states)**.** *The set of reachable states of the flow-sensitive security model as in (3.1) is the minimal set of states* RS *containing* s_{init} *and such that if* $s \in RS$ *and* $s \xrightarrow{\Phi}_\diamond s'$, *for some* Φ, *then* $s' \in RS$.

We have defined general notions related to the syntax and operational semantics of a flow-sensitive security model. It is then straightforward to capture the basic notion of security across the federated cloud.

[1] Note that $\leq$ denotes multiset inclusion.
[2] Note that '$-$' and '$+$' denote multiset subtraction and addition, respectively.

Definition 23. *A state s is* secure *if, for every actual entity (e, c, l) present in s, $l \leq_{sec} l(c)$. A flow-sensitive security model as in (3.1) is* secure *if all its reachable states are secure.*

Intuitively, a state is secure if all copies of entities present in the state reside in clouds without causing security violation. One can formulate a general security policy guaranteeing the security of a flow-sensitive security model. Such a policy is formulated by placing a suitable condition on the security guards present in the model.

Theorem 1. *Let FSSM be a flow-sensitive security model as in (3.1) such that, for every action ϕ as in (3.2), $\{c_{k+1}, \ldots, c_{k+m}\} \subseteq \{c_1, \ldots, c_k\}$ and if $(l_1, \ldots, l_{k+m}) \in \Sigma$ then, for every $i = k+1, \ldots, k+m$:*

$$\coprod_{\{p>k|c_p=c_i\}} l_p \quad \leq_{sec} \quad \prod_{\{r\leq k|c_r=c_i\}} l_r \ . \tag{3.3}$$

Then FSSM is secure provided that its initial state is secure.

Proof: Let $l_1, \ldots, l_{k+m} \in L_{sec}$, if $p > k$ and $r \leq k$, according to (3.2), we know l_p and l_k represent the security levels of the entities which are associated with ϕ_σ^{out} and ϕ_σ^{in}, respectively.

Note that $c_p = c_i$ and $c_r = c_i$ so it implies that $c_p = c_r$. Then we have $l_p \leq_{sec} l_r$, i.e., if two entities are allocated on the same cloud, the security levels of the entities which are associated with ϕ_σ^{out} has lower security status than ϕ_σ^{in}. Therefore, we can say the least upper bound $\coprod$ of l_p is lower or equal than the greatest lower bound $\prod$ of l_r, then we can obtain (3.3).

□

The above result can only be applied in specific cases; in general, we need to verify that a given system specification yields a secure system. In the next section, we will outline how coloured Petri nets can be used to provide a suitable analytical tool.

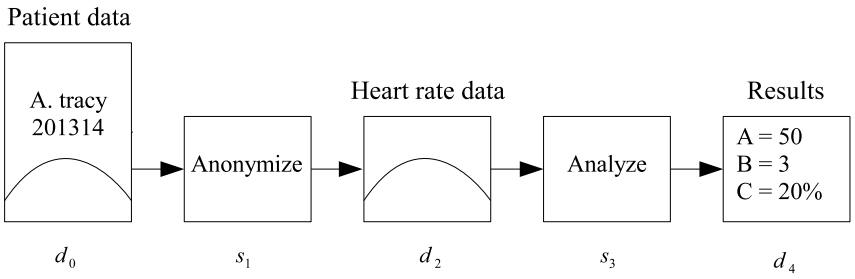


Figure 3.1: The medical research application example from [18].

3.3 Coloured Petri Nets Model

Coloured Petri Nets (CPNs) can be used to model concurrent systems, including information flow in FCSs. In particular, Petri net theory provides powerful analysis techniques which can be used to verify the correctness of workflow procedures [22]. Transitions in Petri nets can be interpreted as occurrences of events or executions of tasks in a system. In our representation of FCSs in CPNs, the relations between events are explicit, and the representation of the system can alleviate the state explosion problem [23].

CPNs allow one to model multi-type cases in a process specification [12], which can model different entities of an FCS. Each token can carry complex information and/or data. Arc expressions and multiple exits can be used to model the various workflow logics. Moreover, guard functions associated with transitions can be used to specify security-related conditions.

3.3.1 Case Study

As a case study adapted to our purposes, we will use a simplified version of the federated cloud system of [18]. It is a medical research application in which data from a set of patients' heart rate monitors is analysed, as illustrated in Figure 3.1.

Informally, the process can be described as follows:

- The input data (d_0) is a file with a header identifying the patient (name and patient number), followed by a set of heart rate data recoded over a period of time;

- A service (s_1) strips off the header, leaving only the heart rate data (d_2);

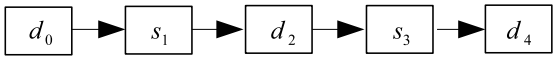


Figure 3.2: A workflow of the case study in [18].

Table 3.1: Security level of clouds, services and data in the FCS of [18].

Clouds	Security level	Services	Security level	Data	Security level
c_0	0	s_1	1	d_0	1
c_1	1	s_3	0	d_2	0
				d_4	0

- A second service (s_3) analyzes the heart rate data, and produces results (d_4).

Analyzing the heart rate data (s_3) is costly and would benefit from a cheap, scalable resources that are available on public clouds. However, considering that storing medical records on a public cloud can breach confidentiality, some organizations prefer to deploy the whole workflow on a secure private cloud. Such a policy may overstretch the limited resources available on the private cloud, resulting in degraded performance and negative impact on other applications. To address this problem, the partitioning of the application between a private cloud and a public cloud could provide a better solution.

In our case study, we use two clouds, one public cloud c_0 and one private cloud c_1. The proposed workflow operates on sensitive medical data processed on the private cloud, and anonymised data that can be deployed on the public cloud.

Figure 3.2 is a workflow which is derived from Figure 3.1, with the security settings shown in Table 3.1, where $\{c_0, c_1\}$ are the clouds in the system, $\{s_1, s_3\}$ are the services, and $\{d_0, d_2, d_4\}$ are the data.

After determining valid mappings of services and data to clouds based on the Bell-Lapadula rules, the workflow in Figure 3.2 leads to six valid partitionings. The partitioning chosen for this case study is shown in Figure 3.3: $\{s_1, d_0, d_2\}$ are

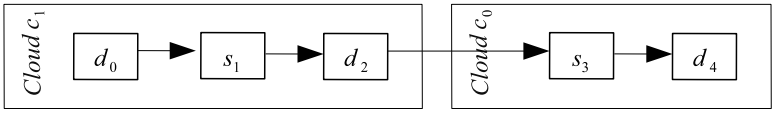
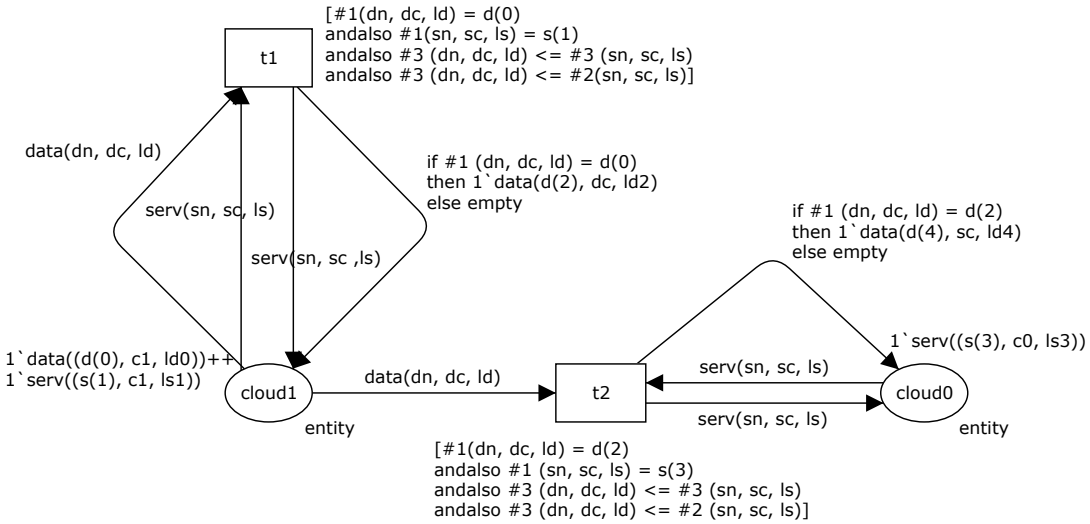


Figure 3.3: A valid workflow partitioning for the FCS of [18].

services and data deployed on cloud c_1 whose security level is 1, while $\{s_3, d_2, d_4\}$ are deployed on cloud c_0 whose security level is 0.

3.3.2 CPN model



· val $ls_1 = 1$;	· closet $lsec$ = int with 0..1;
· val $ls_3 = 0$;	· closet dName = index d with 0..4;
· val $ld_0 = 1$;	· closet dataInfor = product dName * lsec * lsec ;
· val $ld_2 = 0$;	· colset sName = index s with 1..3 ;
· val $ld_4 = 0$;	· colset servInfor = product sName * lsec * lsec ;
· val $c_1 = 1$;	· colset entity = union data: dataInfor + serv: servInfor;
· val $c_0 = 0$;	· var sn: sName;
· var dn: dName;	· var ld, ls, dc, sc: lsec;

Figure 3.4: CPN model (including the colour sets) for a federated cloud system.

Based on the FCS model outlined above, we built a CPN model capturing the information flow of the FCS in the case study, shown in Figure 3.4.

The model provides an abstract view of CPN model of the chosen workflow partitioning of the FCS. It has two places *cloud*1 and *cloud*0, which represent the two clouds, and two transitions, t_1 and t_2. Places store the entities of the system, which have the type *entity* as the colour set (describing the data are needed to be

processed in the system). The declarations of the colour sets in Figure 3.4 tell us that *entity* is a union type, which corresponds to a datatype in Standard ML, with the values being either data (*data*) or services (*serv*). *data* and *serv* have an associated product colour sets which allows one to distinguish each individual entity. Thus, the type *entity* contains the values $\{data((d(0), 1, 1), serv(s(1), 1, 1), \dots\}$. Each token has a colour which has three fields. The first field is an element of $dName$ or $sName$, and thus it is d_0 or d_1 or d_2 or d_3 or d_4 or s_1 or s_2 or s_3, which specifies the name of the data/services in the system. The second element is an integer which specifies the security level of the cloud where the service/data is located. The third element is also an integer, specifying the security level of the service/data. Each of the two transitions, t_1 and t_2, represents a move from one state to the next.

3.4 Opacity in Cloud Computing Systems

Observing behaviour patterns of users can lead to leakages of secure information. Information sharing means that the behaviour of one cloud user may appear visible to other cloud users or adversaries, and observations of such behaviours can potentially help adversaries to build covert channels. Opacity is a uniform approach for describing security properties expressed as predicates [16; 24; 25]. A predicate is opaque if an observer of the system is unable to determine the truth of the predicate in a given run of the system. In this section, we will discuss one of the versions of opacity in the context of workflows executed on federated cloud computing systems.

Let $FSSM = (\mathcal{A}, s_{init})$ be a flow-sensitive security model as in (3.1). A run of $FSSM$ is a finite sequence

$$\xi = \Phi_1 \dots \Phi_n \quad (n \geq 0) \tag{3.4}$$

such that there are states $s_{init} = s_1, \dots, s_{n+1}$ satiafying $s_i \xrightarrow{\Phi_i}_{\diamond} s_{i+1}$, for $i = 1, \dots, n$. The set of all runs of $FSSM$ will be denoted by $RUN(FSSM)$.

To model the different capabilities for observing the system modelled by

FSSM one can use *observation functions*:

$$obs : RUN(FSSM) \to Obs^* \tag{3.5}$$

where Obs be a set of *observables*. In what follows, we consider a *state observation function* obs for which there is a map obs' associating $obs'(\Phi) \in Obs \cup \{\epsilon\}$ with every Φ as in Definition 21, in such a way that

$$obs(\xi) = obs'(\Phi_1) \dots obs'(\Phi_n) \tag{3.6}$$

for every run $\xi = \Phi_1 \dots \Phi_n$ in $RUN(FSSM)$.

Given the observation function obs, we are now interested in whether an observer can establish a property γ (a predicate over system runs) for a run of *FSSM* having only access to the result of the observation function. As one can identify γ with its characteristic set, i.e. the set of all those runs for which it holds, we would want to find out whether the fact that the underlying run belongs to $\gamma \subseteq RUN(FSSM)$ can be deduced by the observer on the basis of an observed execution of the system. Moreover, we are interested in the *final* opacity predicate γ_Z, defined as the set of all the runs ξ as in (3.4) satisfying $s_n \in Z$, for some set of states Z [16]. Intuitively, this means that we are interested in finding out whether an observed run of the system represented by *FSSM* ended in one of secret (or sensitive) states belonging to Z. Note that we are not interested in establishing whether the underlying run does not belong to γ; to do this, we would consider the property $\bar{\gamma} = RUN(FSSM) \setminus \gamma$.

We then say that γ is *opaque* w.r.t. obs if, for every run $\xi \in \gamma$, there is another run $\xi \in \bar{\gamma}$ such that $obs(\xi) = obs(\xi')$. In other words, if all runs in γ are covered by runs in $\bar{\gamma}$:

$$obs(\gamma) \subseteq obs(\bar{\gamma}) \ .$$

3.4.1 Case Study

The scenario of this case study involves three clouds (W, X, and Y) and a number of processes, which all together form an e:*shop* application. Cloud X hosts a web portal e:*win*; Cloud W hosts two providers, e:*prov1* and e:*prov2*;

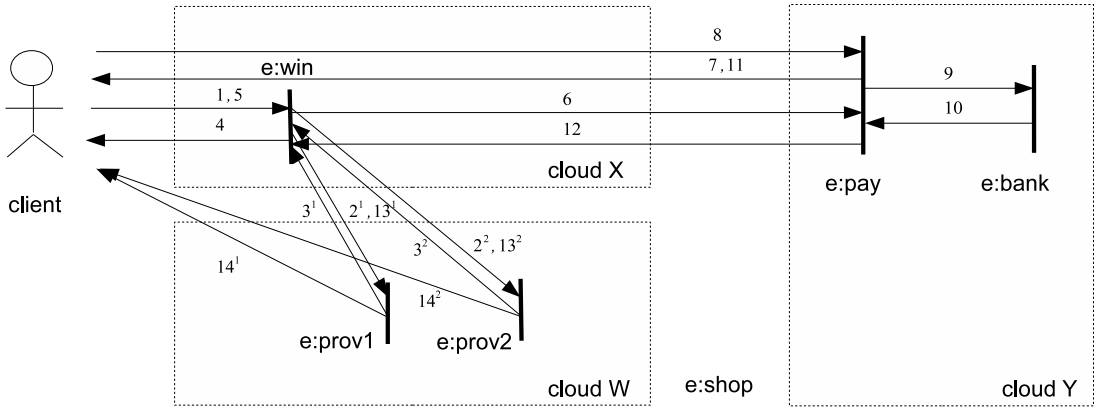


Figure 3.5: Information flow in a cloud based *e*:*shop*.

and cloud Y hosts a payment handling site, *e*:*pay*, and an internet bank, *e*:*bank*. The functionality of the *e*:*shop* is built around the transmission of information between the various predefined process participants, such as the providers and bank.

Figure 3.5 shows the basic structure of the execution scenario for the *e*:*shop*. The role of each of the processes is as follows:

- *client* tries to buy a product online, accessing to cloud based retail applications through a web browser.
- *e*:*win* is a web portal which provides a platform for trading.
- *e*:*prov1* and *e*:*prov2* supply products traded by *e*:*shop*.
- *e*:*pay* is an agent through which *client* can make online payment and transfer money between bank accounts.
- *e*:*bank* can handle payments and deposits through the *e*:*pay* agent.

The generic behaviour of *e*:*shop* is shown in Table 3.2. It starts with a purchase request sent from *client* to *e*:*win*. The request is forwarded to the product providers, *e*:*prov1* and *e*:*prov2*, who reply with the relevant product information. *e*:*win* forwards the received information to *client*, who selects one of the two products and sends the decision back to *e*:*win*. Then *e*:*win* forwards the decision to *e*:*pay* which contacts *client* asking for payment details. After receiving

Table 3.2: The sequence of interactions between the components of the *e*:*shop* system.

	Entities (subject)	Actions	Sender	Receiver (object)
1	client's request	Φ_1	client	e:win
2	forward client's request	Φ_2^i	e:win	e:prov$i$ (i=1,2)
3	product's information	Φ_3^i	e:prov$i$ (i=1,2)	e:win
4	forward product's information	Φ_4	e:win	client
5	choose products	Φ_5	client	e:win
6	forward the client to the payment	Φ_6	e:win	e:pay
7	client's information request	Φ_7	e:pay	client
8	make payment	Φ_8	client	e:pay
9	contact bank	Φ_9	e:pay	e:bank
10	make payment	Φ_{10}	e:bank	e:pay
11	send invoices	Φ_{11}	e:pay	client
12	inform of the payment	Φ_{12}	e:pay	e:win
13	inform of the payment	Φ_{13}^i	e:win	e:prov$i$ (i=1,2)
14	ship product	Φ_{14}^i	e:prov$i$ (i=1,2)	client

the payment information, *e*:*pay* contacts *e*:*bank* to carry out the payment. Then *e*:*pay* sends an invoice to *client* and informs *e*:*win*. Finally, *e*:*win* contacts the selected provider to trigger the shipment of the product to *client*. Moreover, the selected provider is not allowed to reveal their identity to an observer.

We also assume that no provider is discriminated against. Moreover, messages communicated between the clouds X, W and Y are invisible, and other messages are visible. However, the observer has no means of detecting their content (but can observe the specific cloud from which a message originated or was sent to). This can be captured by the following (static) observation function:

$$
\begin{array}{lllllllll}
obs'(\Phi_1) & = & a & obs'(\Phi_2^1) & = & \epsilon & obs'(\Phi_2^2) & = & \epsilon \\
obs'(\Phi_3^1) & = & \epsilon & obs'(\Phi_3^2) & = & \epsilon & obs'(\Phi_4) & = & b \\
obs'(\Phi_5) & = & a & obs'(\Phi_6) & = & \epsilon & obs'(\Phi_7) & = & c \\
obs'(\Phi_8) & = & d & obs'(\Phi_9) & = & \epsilon & obs'(\Phi_{10}) & = & \epsilon \\
obs'(\Phi_{11}) & = & c & obs'(\Phi_{12}) & = & \epsilon & obs'(\Phi_{13}^1) & = & \epsilon \\
obs'(\Phi_{13}^2) & = & \epsilon & obs'(\Phi_{14}^1) & = & e & obs'(\Phi_{14}^2) & = & e
\end{array}
$$

Using opacity, we may show that visible interaction do not reveal the identity of the provider supplying the goods. To see this, we consider a property γ consisting of all execution scenarios where the first provider supplied the goods, i.e., executions of the following form:

$$\begin{aligned}
\xi_1 &= \Phi_1\Phi_2^1\Phi_2^2\Phi_3^1\Phi_3^2\Phi_4\Phi_5\Phi_6\Phi_7\Phi_8\Phi_9\Phi_{10}\Phi_{11}\Phi_{12}\Phi_{13}^1\Phi_{14}^1 \\
\xi_2 &= \Phi_1\Phi_2^2\Phi_2^1\Phi_3^1\Phi_3^2\Phi_4\Phi_5\Phi_6\Phi_7\Phi_8\Phi_9\Phi_{10}\Phi_{11}\Phi_{12}\Phi_{13}^1\Phi_{14}^1 \\
\xi_3 &= \Phi_1\Phi_2^1\Phi_2^2\Phi_3^2\Phi_3^1\Phi_4\Phi_5\Phi_6\Phi_7\Phi_8\Phi_9\Phi_{10}\Phi_{11}\Phi_{12}\Phi_{13}^1\Phi_{14}^1 \\
\xi_4 &= \Phi_1\Phi_2^2\Phi_2^1\Phi_3^2\Phi_3^1\Phi_4\Phi_5\Phi_6\Phi_7\Phi_8\Phi_9\Phi_{10}\Phi_{11}\Phi_{12}\Phi_{13}^1\Phi_{14}^1
\end{aligned}$$

The set of observations they generate is given by $obs(\gamma) = \{abacdce\}$. We then note that $\bar{\gamma}$ comprises, among others, executions of the following kind:

$$\begin{aligned}
\xi_1 &= \Phi_1\Phi_2^1\Phi_2^2\Phi_3^1\Phi_3^2\Phi_4\Phi_5\Phi_6\Phi_7\Phi_8\Phi_9\Phi_{10}\Phi_{11}\Phi_{12}\Phi_{13}^2\Phi_{14}^2 \\
\xi_2 &= \Phi_1\Phi_2^2\Phi_2^1\Phi_3^1\Phi_3^2\Phi_4\Phi_5\Phi_6\Phi_7\Phi_8\Phi_9\Phi_{10}\Phi_{11}\Phi_{12}\Phi_{13}^2\Phi_{14}^2 \\
\xi_3 &= \Phi_1\Phi_2^1\Phi_2^2\Phi_3^2\Phi_3^1\Phi_4\Phi_5\Phi_6\Phi_7\Phi_8\Phi_9\Phi_{10}\Phi_{11}\Phi_{12}\Phi_{13}^2\Phi_{14}^2 \\
\xi_4 &= \Phi_1\Phi_2^2\Phi_2^1\Phi_3^2\Phi_3^1\Phi_4\Phi_5\Phi_6\Phi_7\Phi_8\Phi_9\Phi_{10}\Phi_{11}\Phi_{12}\Phi_{13}^2\Phi_{14}^2
\end{aligned}$$

Hence $obs(\gamma) \subseteq obs(\bar{\gamma})$, and so γ is an opaque property. As a result, it is never possible to say for sure that it was the first provider who supplied the goods. Since the argument is completely symmetric, we can conclude that the identity of providers is kept secret.

3.5 Probabilistic Behaviour of Opacity in Cloud Computing Systems

An observation cannot establish a predicate, if for any run of the system in which the predicate is true, there is a run for which the predicate is false, and the two runs are observationally equivalent under the defined observation function. However, in the case where the probability of the first run is significantly higher than the probability of the second, the observer may have good reasons to believe that the predicate is none the less true.

Consider that a malicious service provider (insider) can observe any specific

users' behaviours and interactions with the server by studying the patterns of users behaviours in the cloud system, and then the malicious insider can deduce information about the users, which might cause confidential information leak. Here, we consider the probabilistic opacity in the cloud computing systems which allows us to reason the quantitative properties of the systems.

All the resources, the computation technology and the software required are located in the server system refereed to as the environment. Users request data/service and required processing software from computing system through a device with internet connection. After all the data processing is done at the computing system, the result is sent back to the client. For the purpose of measuring the opacity of the system, we consider the probability distributions on random variables of user behaviour traces.

We generally use the language of random variable rather than talk directly about distributions. Here, D is a finite set with a discrete probability distribution. A discrete random variable X is a surjective function which maps events to values of a countable set (e.g., the integers), with each value in the range having probability greater than zero, and $\mathcal{R}$ is the finite range of X, i.e. $X : D \to \mathcal{R}(D)$. For each $d \in D$, we write $p(d)$ for its probability, then we have:

$$0 < p(d) \leq 1, \quad and \quad \sum_{d \in D} p(d) = 1$$

The system can be considered as a communication channel, discrete random variable X is constructed to model a finite set of possible traces performed by the end users during their interactions with the system. Each trace is a finite sequence of actions performed by the user, i.e., $\xi = \Phi_1 \dots \Phi_n, (n \in N)$, which express how the user reaches the finial action Φ_n from the initial action Φ_1 during requesting a specific service. In addition, each trace is associated with a positive probability, and the sum of the probabilities of all possible traces is 1. In order to consider probabilistic behaviours and quantitative analysis of opacity in the system, we consider that for all traces in the system:

$$\sum_{i=1}^{n} p(\xi_i) = 1$$

In this study, we require the traces of system should be finite, thus, $n \in N$ denotes the number of the finite traces of the system.

Following the definitions of *observation function* in Section 3.4. Therefore, the observed traces form a distribution:

$$\sum_{i=1}^{m} p(obs(\xi_i)) = 1$$

where $m \in N$ denotes the number of the finite observable traces.

The computing system is modelled as a tuple which includes: a set of users interacting with the system; a set of actions and observables during the interactions between the users and the system; a set of actual traces to model the sequences of actions performed by the users during interacting with the system; a set of observed traces to model the observations upon the actual traces; random variables built on both of the actual traces and the observed traces. It assumes that the random variables on the traces are obtained by repeated experiments.

Definition 24. Service-based computing system *is a tuple:*

$$(\mathcal{U}, \mathcal{A}, Obs, \mathcal{T}, \mathcal{O}, \mu_{\mathcal{T}}, \mu_{\mathcal{O}})$$

- $\mathcal{U}$ is a finite set of *users*;
- $\mathcal{A}$ is a finite set of *actions* modelling the interactions between a server and the users.
- *Obs* is a finite set of *observables*;
- $\mathcal{T}$ is a set of *actual traces*, i.e., possible sequences of actions of all the users;
- $\mathcal{O}$ is a finite set of *observed traces* obtained from the actual traces;
- $\mu_{\mathcal{T}}$ represents a family of probability distributions, each of which is on all actual traces of each client interacting with the server.

  For any specific user $u \in \mathcal{U}$, let all possible traces of a user u requesting a service s located in the system be denoted by: $\mathcal{T}_{u,s} = \{\xi_i \mid 1 \leq i \leq n_{u,s}\}$, where $\mathcal{T}_{u,s} \subseteq \mathcal{T}$. The probability distribution: $P_{u,s} : T_{u,s} \to [0, 1]$, and

$\forall u \in \mathcal{U}$, $\sum_{i=1}^{n_{u,s}} p(\xi_i) = 1$, where $P_{u,s} \in \mu_{\mathcal{T}}$, and $T_{u,s}$ denotes the random variable of $\mathcal{T}_{u,s}$ under the condition of user u requesting the service s;

- $\mu_{\mathcal{O}}$ denotes a family of probability distributions, each of which is on all observed traces obtained from the actual traces.

3.5.1 Threat Model

Information lattice is proposed for the entities in the cloud computing system in Section 3.1. In this part, we assume that there are two types of actions: *hidden* actions and *observable* actions, which related to *high* security level and *low* security level respectively. Actions labelled *hidden* are confidential and hidden to the adversaries, and actions labelled *observable* are public and observable to the adversaries. The classification can be based on the security preserving mechanisms or policies applied in the computing system. Therefore, the actions can be described as the union of two disjoint sets:

$$\mathcal{A} = high \uplus low$$

For each trace, some part of it is hidden and some part of it is observable, which makes some traces equivalent to the others when only considering observable actions. Therefore, adversaries can derive some confidential information by building sets of equivalence classes from the observations.

3.5.2 Actual Traces and Observations

We now look at the properties of observations on the behaviours of users within the computing system through equivalence relations. Consider that an end user logs into a service provider of the system, and performs a sequence of actions denoted by $(\Phi_j \mid 1 \leq j \leq n, n \in N, \Phi_j \in \mathcal{A})$. The finite sequence of actions of a user forms a trace which starts from the user login in and terminates at the user login out. The service-based computing works on a client-server basis and provides server-based applications and all data services to the user.

We consider that the possible actual traces of each end user are viewed as a random variable and follow a probability distribution, each trace is a sequence of

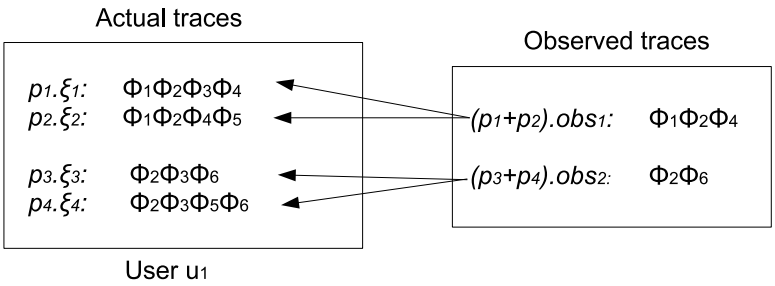


Figure 3.6: An example for observations on actual traces

actions, i.e., $\xi_i = (\Phi_j \mid \Phi_j \in \mathcal{A})$, and $\xi_i \in \mathcal{T}$. Let the probability of a trace ξ_i be $p(\xi_i) = p_i$, and $\sum_{i=1}^{m} p_i = 1$. Let Φ_1 denotes login in the system, and Φ_n denotes login out. The observable traces can be viewed as projections of the actual traces.

Example 6. Consider Figure 3.6 as an example. User u_1 performs four possible actual traces $\{\xi_i \mid i \in \{1, 2, 3, 4\}\}$. Assuming the probability of trace ξ_i is $p(\xi_i) = p_i$, we have:

- ξ_1: $(\Phi_1\Phi_2\underline{\Phi_3}\Phi_4) \rightarrow p_1$
- ξ_2: $(\Phi_1\Phi_2\Phi_4\underline{\Phi_5}) \rightarrow p_2$
- ξ_3: $(\Phi_2\underline{\Phi_3}\Phi_6) \rightarrow p_3$
- ξ_4: $(\Phi_2\underline{\Phi_3\Phi_5}\Phi_6) \rightarrow p_4$

Moreover, $\sum_{i=1}^{4} p_i = 1$. The actual traces are obfuscated due to the applied security policy of the system which produces observable traces. Assuming that underlined Φ_3, Φ_5 actions are labelled as *high* security, we obtain the following observations from the attackers' point of view:

- $\delta_1 : (\Phi_1\Phi_2\Phi_4) \rightarrow p(\delta_1)$
- $\delta_2 : (\Phi_2\Phi_6) \rightarrow p(\delta_2)$

and $\mathcal{O} = \{\delta_1, \delta_2\}$, i.e., the observation functions can be summarised as:

- $obs(\xi_1) = obs(\xi_2) = \delta_1$
- $obs(\xi_3) = obs(\xi_4) = \delta_2$

The probability of observation δ_1 is $p(\delta_1) = p_1 + p_2$, and the probability of observation δ_2 is $p(\delta_2) = p_3 + p_4$. The adversary therefore builds equivalence relations on the inverse images of the observations and derives information from the observations on users behaviours in the system. □

Definition 25. Equivalence classes *In mathematics, when a set has an equivalence relation defined on its elements, there is a natural grouping of elements that are related to one another, forming what are called equivalence classes.*

In this example, the equivalence classes are based on the security type of actions of traces. Intuitively, we consider the observation as the sum of the projection on the actual traces. Information on the users' behaviours from the traces can be partially deduced from these observations. Under repeated observation on traces, we can deduce the probability distribution for the projections of the possible traces for end users.

3.6 Quantitative Analysis of Opacity in Cloud Computing Systems

In this section, quantitative analysis of opacity in the cloud computing systems will be introduced.

3.6.1 Observational Equivalence: π-opacity

If ξ_1 and ξ_2 are two actual traces, such that the observations of them are equal, i.e., $obs(\xi_1) = obs(\xi_2)$, we write $\xi_1 \backsimeq \xi_2$ and say that they are observationally equivalent. Different security policy might produce different observations for the same actual traces of the end user. For all possible actual traces, if observations on them are the same, then the observer can not tell any difference of the user's actual behaviours, and the system is completely secure under its security policy.

We can say predicate γ is π-*opaque* w.r.t. *obs* if the probability of having a run in a predicate γ which is not covered by a run in $\bar{\gamma}$ is zero, i.e., $p(\gamma \backslash obs^{-1}(obs(\bar{\gamma}))) = 0$.

Example 7. Consider that a user u requests a service s, and assume that the actual traces are $\mathcal{T}_{u,s} = \{\xi_1, \xi_2, \xi_3, \xi_4\}$, where:

- $\xi_1 = (\Phi_1 \underline{\Phi_2} \Phi_3 \underline{\Phi_4} \Phi_5) \rightarrow \frac{1}{4}$
- $\xi_2 = (\Phi_1 \Phi_3 \underline{\Phi_4} \Phi_5) \rightarrow \frac{1}{2}$
- $\xi_3 = (\Phi_1 \underline{\Phi_2} \Phi_3 \Phi_5) \rightarrow \frac{1}{8}$
- $\xi_4 = (\Phi_1 \Phi_3 \Phi_5) \rightarrow \frac{1}{8}$

We have:

$$obs(\xi_1) = obs(\xi_2) = obs(\xi_3) = obs(\xi_4) = (\Phi_1 \Phi_3 \Phi_5) \rightarrow 1$$

i.e., $\xi_1 \backsimeq \xi_2 \backsimeq \xi_3 \backsimeq \xi_4$. The above implies that the traces are observantly equivalent to each other and we say that γ is π-*opaque*. □

3.6.2 Proportion-based Opacity Measurement: π_{χ_p}-opacity

The observational equivalence is too strict in practice. One might require only that the probability of the non-equivalent traces is low. To capture this, we first define the opacity quantity as the proportion of the equivalent traces over the whole actual traces. The higher such quantity is, the more secure the system becomes. This also meets our intuition: lower probability of non-equivalent traces means that it is harder for the observers to tell the differences in users' behaviours.

Definition 26. Proportion-based opacity quantity *is the probability of the set of the equivalent traces:*

$$\chi_p = p(\{\ \xi \in \mathcal{T} | \forall \xi' \in \mathcal{T} \backslash \{\xi\} : obs(\xi) \neq obs(\xi')\})$$

The opacity is maximum when χ_p is 0, because the observer can not identify the differences of observational traces. On the contrary, the opacity is minimum when χ_p is 1. We say γ is π_{χ_p}-opaque.

Example 8. Consider the actual traces of user u requesting service s are $\mathcal{T}_{u,s} = \{\xi_1, \xi_2, \xi_3, \xi_4\}$, where:

- $\xi_1 = (\Phi_1 \underline{\Phi_2 \Phi_3} \Phi_4 \Phi_6) \rightarrow \frac{1}{8}$
- $\xi_2 = (\Phi_1 \underline{\Phi_2} \Phi_4) \rightarrow \frac{1}{4}$
- $\xi_3 = (\underline{\Phi_2 \Phi_3} \Phi_4 \underline{\Phi_5}) \rightarrow \frac{1}{8}$
- $\xi_4 = (\Phi_1 \underline{\Phi_3} \Phi_4 \underline{\Phi_5}) \rightarrow \frac{1}{2}$

The actions Φ_2, Φ_3, Φ_5 are non-observables. Therefore, the observation function gives:

- $obs(\xi_1) = (\Phi_1\Phi_4\Phi_6) \rightarrow \frac{1}{8}$
- $obs(\xi_2) = obs(\xi_4) = (\Phi_1\Phi_4) \rightarrow \frac{3}{4}$
- $obs(\xi_3) = (\Phi_4) \rightarrow \frac{1}{8}$

Hence ξ_2 and ξ_4 are equivalent traces. Therefore, the proportion-based opacity measurement is as follows:

$$\chi_p = p(\xi_1) + p(\xi_3) = \frac{1}{8} + \frac{1}{8} = 0.25$$

therefore, γ is $\pi_{0.25}$-opacity. □

3.6.3 Entropy-based Opacity Measurement: $\bar{\pi}_{\mathcal{I}}$-opacity

The measurement we consider here is based on the likelihood of a particular trace of actions being performed by the end user, i.e., on the probability distribution on the user's behaviours to measure the observations. Shannon's measures are based on a logarithmic measure of the surprise, inherent in a probabilistic event. Therefore, it is natural to consider Shannon's entropy as the basis of the information loss measurement.

We consider possible traces of an end user interacting with a service provider in the system as random variables, and define the quantity of observations of the computing system by using the concept of mutual information between the actual traces and observations.

Definition 27. *Consider any specific user $u \in \mathcal{U}$ requesting service s. The quantity of entropy-based information (uncertainty) loss due to user u in the computing system $(\mathcal{U}, \mathcal{A}, Obs, \mathcal{T}, \mathcal{O}, \mu_{\mathcal{T}}, \mu_{\mathcal{O}})$ is defined as:*

$$\mathcal{I}(T_{u,s}; O_{u,s}) = \mathcal{H}(T_{u,s}) - \mathcal{H}(T_{u,s}|O_{u,s})$$

where $T_{u,s}$ and $O_{u,s}$ denotes the random variables for the user u's actual traces $\mathcal{T}_{u,s}$ and observations traces $\mathcal{O}_{u,s}$ respectively; $\mathcal{H}(T_{u,s})$ is the entropy (uncertainty

measurement) of traces in $\mathcal{T}_{u,s}$, and $\mathcal{H}(T_{u,s}|O_{u,s})$ is the conditional entropy of $T_{u,s}$, given the observation $O_{u,s}$ (the remaining uncertainty $\mathcal{T}_{u,s}$ after observing $\mathcal{O}_{u,s}$). Here, we say γ is $\bar{\pi}_{\mathcal{I}}$-opacity.

Example 9. Consider the example discussed in Figure 3.6. Assume $p(\xi_i) = \frac{1}{4}$, for $i \in \{1, 2, 3, 4\}$. Therefore,

$$obs = \{\delta_1, \delta_2\} \quad \& \quad p(\delta_1) = \frac{1}{2} \quad \& \quad p(\delta_2) = \frac{1}{2}$$

The observations is then computed as follows:

$$\mathcal{I}(\xi_1; \delta_1) = \mathcal{H}(\xi_1) - \mathcal{H}(\xi_1|\delta_1) = 4 * \frac{1}{4} \log_2 4 - (2 * \frac{1}{2} \log_2 2) = 1$$

□

The mutual information between ξ_1 and δ_1 is 1, therefore, the information observation is 0, i.e., all the observation traces are equal to each other under that applied security preserving policy, and we say γ is $\bar{\pi}_0$-opacity.

The above discussion focuses on the case of one user. The following definition defines the security measurement for all users in the computing system. It is based on the mean value of the security measurements of all users' behaviours, each of which is calculated by using Definition 27.

Definition 28. *The quantity of the information loss of a computing system is defined as the mean value of all users' information loss:*

$$\overline{\chi_e} = \sum_{i=1}^{m} p_i \cdot \mathcal{I}(T_i; O_i)$$

where m is the number of end users, p_i denotes the probability of all possible traces of user u_i requesting a service s, T_i and O_i denote the random variables of actual traces and observed traces under condition of user u_i defined by $\mu_{\mathcal{T}}$ and $\mu_{\mathcal{O}}$, $\mathcal{I}(T_i; O_i)$ denotes the information loss of user u_i.

3.6.4 Channel Capacity: $\hat{\pi}_C$-opacity

The opacity measurement in previous section is calculated with respect to the average of each user's observation measurement. For some cases, we might need to provide the maximum one among all users' information loss. We next define another measurement based on the concept of channel capacity.

Let us consider the security preserving mechanisms of the system providing secure communication channels for clients and services in the information theoretical sense. A channel consists of a set of inputs (actual traces $\mathcal{T}$), a set of outputs (observed traces $\mathcal{O}$), and a set of images between them ($F : T \rightarrow O$), where T and O denotes the random variables of $\mathcal{T}$ and $\mathcal{O}$. Intuitively, actual behaviours of the users can be partially derived by the observers through the observed traces.

The maximum mutual information between T and O over all possible users $u \in \mathcal{U}$ regarding to requesting a service s is considered as the channel's capacity.

Definition 29. The channel capacity of computing system *is defined as:*

$$C = max_{u \in \mathcal{U}} \mathcal{I}(T_{u,s}; O_{u,s})$$

when all the user u request a service s. We say γ is $\hat{\pi}_C$-opacity.

Example 10. Let us consider a simple example: an on-line paper assessing management system called *iWorks*. The system *iWorks* allows the users to browse papers, contribute reviews, and discussion through the Web. We assume the follows:

- The possible actions are: $\mathcal{A} = \{\Phi_i | 1 \leq i \leq 10\}$ that include: logging in (Φ_1), downloading papers (Φ_2), uploading papers (Φ_3), reviewing papers (Φ_4), commenting papers (Φ_5), deleting papers (Φ_6), ranking papers (Φ_7), edit personal information (Φ_8), sending private comments to authors (Φ_9), logging out (Φ_{10});
- Actions Φ_1 to Φ_6 and Φ_{10} are observable behaviours, and actions Φ_7, Φ_8, Φ_9 are hidden behaviours according to the security policy applied by the system: $Obs = \{\Phi_1, \Phi_6, \Phi_{10}\}$;

- There are two users: $\mathcal{U} = \{u_1, u_2\}$;
- The *iWorks* service requested by the users is denoted by s;
- The sum of the probability of all traces performed by all the users is 1. The probability of all possible traces of user u_1 is $\frac{2}{3}$, and that of user u_2 is $\frac{1}{3}$;
- The possible actual traces of each user and conditional probabilities of each user's traces are as follows:

user	actual traces	cond. prob. under u_i
$\frac{2}{3}.u_1$	$\xi_{11} : \Phi_1\ \Phi_2\ \Phi_4\ \Phi_5\ \underline{\Phi_7}\ \underline{\Phi_9}\ \Phi_{10}$	$\frac{1}{4}$
	$\xi_{12} : \Phi_1\ \Phi_3\ \underline{\Phi_8}\ \Phi_{10}$	$\frac{1}{6}$
	$\xi_{13} : \Phi_1\ \underline{\Phi_8}\ \Phi_2\ \underline{\Phi_9}\ \Phi_4\ \Phi_5\ \underline{\Phi_7}\ \Phi_{10}$	$\frac{1}{6}$
	$\xi_{14} : \Phi_1\ \underline{\Phi_9}\ \Phi_2\ \Phi_4\ \Phi_5\ \underline{\Phi_7}\ \Phi_{10}$	$\frac{1}{6}$
	$\xi_{15} : \Phi_1\ \underline{\Phi_9}\ \Phi_3\ \Phi_4\ \underline{\Phi_7}\ \Phi_{10}$	$\frac{1}{8}$
	$\xi_{16} : \Phi_1\ \underline{\Phi_8}\ \Phi_3\ \Phi_4\ \Phi_{10}$	$\frac{1}{8}$
$\frac{1}{3}.u_2$	$\xi_{21} : \Phi_1\ \Phi_2\ \Phi_6\ \underline{\Phi_8}\ \Phi_{10}$	$\frac{1}{2}$
	$\xi_{22} : \Phi_1\ \Phi_2\ \Phi_6\ \underline{\Phi_7}\ \underline{\Phi_9}\ \Phi_{10}$	$\frac{1}{4}$
	$\xi_{23} : \Phi_1\ \Phi_2\ \underline{\Phi_8}\ \underline{\Phi_9}\ \Phi_6\ \Phi_{10}$	$\frac{1}{4}$

The actions with underline are hidden by the security mechanisms applied by the system. The observations are therefore presented as follows:

user	actual traces	cond. prob. under u_i
$\frac{2}{3}.u_1$	$obs(\xi_{11}) : \Phi_1\ \Phi_2\ \Phi_4\ \Phi_5\ \Phi_{10}$	$\frac{7}{12}$
	$obs(\xi_{12}) : \Phi_1\ \Phi_3\ \Phi_{10}$	$\frac{1}{6}$
	$obs(\xi_{13}) : \Phi_1\ \Phi_3\ \Phi_4\ \Phi_{10}$	$\frac{1}{4}$
$\frac{1}{3}.u_2$	$obs(\xi_{21}) : \Phi_1\ \Phi_2\ \Phi_6\ \Phi_{10}$	$\frac{3}{4}$
	$obs(\xi_{22}) : \Phi_1\ \Phi_2\ \Phi_6\ \Phi_{10}$	$\frac{1}{4}$

The entropy-based information loss due to the behaviour of user u_1 is:

$$\mathcal{I}(T_1; O_1) = \mathcal{H}(\frac{7}{12}, \frac{1}{4}, \frac{1}{6}) = 1.38$$

The entropy-based information loss due to the behaviour of user u_2 is:

$$\mathcal{I}(T_2; O_2) = \mathcal{H}(\frac{3}{4}, \frac{1}{4}) = 0.81$$

The overall information loss (i.e., all users' information loss) of the system is:

$$\overline{\chi_e} = 1.38 * \frac{2}{3} + 0.81 * \frac{1}{3} = 1.19$$

The channel capacity of the system is:

$$C = max_{u \in U} \mathcal{I}(T_{u,s}; O_{u,s}) = 1.38$$

In the above, T_i and O_i denote the random variables of actual traces and observed traces of user u_i ($i \in \{1, 2\}$) respectively. Therefore, the γ is $\hat{\pi}_{1.38}$-opacity. □

Note that different security preserving mechanisms or policies might produce different observations. Given a computing system, it is necessary to study the relations among the security policies applied by the system, from the point view of the security degrees produced. Theorem 2 suggests an ordering on such relations.

Theorem 2. *Let P_1, P_2 be two security preserving policies, and η_1, η_2 be two equivalence classes due to P_1, P_2 respectively. We use T to denote the random variable of actual trace, O_1, O_2 to denote the relevant random variables of observed traces under η_1, η_2 . Then we have:*

$$\eta_1 \subseteq \eta_2 \Rightarrow \mathcal{I}(T; O_1) \geq \mathcal{I}(T; O_2)$$

Intuitively, the above implies that if η_2 is coarser than η_1 then the information loss under η_2 is less than the case under η_1.

Proof: $\eta_1 \subseteq \eta_2$ implies η_2 is coarser than or equal to η_1, from a view of partition by building the equivalence class on the actual traces $\mathcal{T}$, i.e., $\eta_1 \cap \eta_2 = \eta_1$. Therefore, the uncertainty measure of the random variable of actual trace T given condition O_2 is bigger than or equal to that of T given condition O_1, i.e.,

$\mathcal{H}(T|O_2) \geq \mathcal{H}(T|O_1)$, and therefore:

$$\begin{aligned} & \mathcal{I}(T;O_1) - \mathcal{I}(T;O_2) \\ = \; & (\mathcal{H}(T) - \mathcal{H}(T|O_1)) - (\mathcal{H}(T) - \mathcal{H}(T|O_2)) \\ = \; & \mathcal{H}(T|O_2) - \mathcal{H}(T|O_1) \geq 0 \end{aligned}$$

We obtain: $\eta_1 \subseteq \eta_2 \Rightarrow \mathcal{I}(T;O_1) \geq \mathcal{I}(T;O_2)$. □

3.6.5 Distance-based Opacity Measurement: $\tilde{\pi}_{\chi_{d_{KL}}}$-opacity, $\ddot{\pi}_{\chi_{d_{JS}}}$-opacity

The above notions may not distinguish differences between different end users. In this part, we introduce a notion of private measurement which can capture the subtle differences between the observed traces of different users. Intuitively, a possible way to assess such differences could be, e.g., to look at the probability distributions induced by the random variable of the observable traces of two different users, and analyse how similar they are. In our last attempt at a notion of privacy loss measurement, we define χ_d which uses distributed divergence as a way to measure the differences between the distribution of observed traces of any two different end users. We consider two divergence here: the *Kullback-Leibler (KL) divergence* and the *Jensen-Shannon (JS) divergence* [26].

The KL divergence is a non-symmetric measure of the difference between two probability distributions O_1 and O_2, which represent the distribution of the observed traces for two users. Note that, the more the similarity of O_1 and O_2, the smaller the distance between O_1 and O_2, the less privacy information is released regarding to the distance based measurement. The KL-based privacy loss measurement is defined as follows:

Definition 30. *Consider any specific user $u \in U$ requesting service s. The quantity of KL-based security measurement due to user u in the computing system $(\mathcal{U}, \mathcal{A}, Obs, \mathcal{T}, \mathcal{O}, \mu_{\mathcal{T}}, \mu_{\mathcal{O}})$ is defined as the inverse of the KL-distance between the random variables for actual traces and observations:*

$$\chi_{d_{KL}}(T_{u,s} \parallel O_{u,s}) = \frac{1}{D_{KL}(T_{u,s} \parallel O_{u,s})} = \frac{1}{\sum_{i=1}^{n} T(i) \log_2 \frac{T(i)}{O(i)}}$$

where $T_{u,s}$ and $O_{u,s}$ denote the random variables for the user u's actual traces $\mathcal{T}_{u,s}$ and observations $\mathcal{O}_{u,s}$ respectively, D_{KL} denotes the KL-distance, $n \in N$ denotes the size of the actual traces, $T(i)$ denotes the probability distribution of the actual trace, $O(i)$ denotes the probability distribution of the observations. We call γ is $\tilde{\pi}_{\mathcal{L}_{d_{KL}}}$-opacity.

Intuitively, such measurement captures the average of the logarithmic difference between the probabilities T and O, where the average is taken using the probabilities T. Note that the KL-distance is always non-negative, and therefore the KL-based measurement is non-negative as well. In addition, the bigger the distance between the actual traces and observations, the less the information is released to the observers which also meets our intuition.

Note that the KL divergence is not symmetric: the KL from T to O is generally not the same as the KL from O to T. One can consider the symmetrised divergence JS-distance as the metric, which is defined as follows:

Definition 31. *Consider any two users $u \in U$ requesting service s. The quantity of JS-based security measurement for privacy due to user u_1 and u_2 in the computing system $(\mathcal{U}, \mathcal{A}, Obs, \mathcal{T}, \mathcal{O}, \mu_{\mathcal{T}}, \mu_{\mathcal{O}})$ is defined as:*

$$\begin{aligned} & \chi_{d_{JS}}(T_{u,s} \parallel O_{u,s}) \\ = \; & \mathcal{H}(T_{u,s}) \cdot [1 - D_{JS}(T_{u,s} \parallel O_{u,s})] \\ = \; & \mathcal{H}(T_{u,s}) \cdot [1 - \frac{1}{2}(D_{KL}(T_{u,s} \parallel O_{u,s}) + D_{KL}(O_{u,s} \parallel T_{u,s}))] \end{aligned}$$

where $T_{u,s}$ and $O_{u,s}$ denotes the random variables for user u's actual traces $\mathcal{T}_{u,s}$ and observation $\mathcal{O}_{u,s}$ respectively, $D_{JS}(T_{u,s} \parallel O_{u,s})$ denotes the JS-distance between $T_{u,s}$ and $O_{u,s}$, and $\mathcal{H}(T_{u,s})$ denotes the entropy for distribution $T_{u,s}$. We call γ is $\ddot{\pi}_{\chi_{d_{JS}}}$-opacity.

The JS-based measurement can be interpreted as the capacity of the information channel with two inputs giving the output distributions T and O. Note that

the JS-distance D_{JS} is always bounded by $[0, 1]$, the JS-based leakage measurement is therefore always bounded by $[0, \mathcal{H}(T_{u,s})]$. Furthermore, this meets our intuition:

- the $\chi_{d_{JS}}(T_{u,s} \parallel O_{u,s})$ is minimum (0) when the distance $D_{JS}(T_{u,s} \parallel O_{u,s})$ is maximum (1);
- the $\chi_{d_{JS}}(T_{u,s} \parallel O_{u,s})$ is maximum ($\mathcal{H}(T_{u,s})$), i.e., the total information entropy of the random variable of actual traces when the distance $D_{JS}(T_{u,s} \parallel O_{u,s})$ is minimum (0).

Similar to definition 28, we can calculate the total quantity of information loss of a computing system by computing the mean value of all users information loss:

$$\chi_{d_{KL}} = \sum_{i=1}^{m} p_i \cdot \chi_{KL}(T_i \parallel O_i) \quad or \quad \chi_{d_{JS}} = \sum_{i=1}^{m} p_i \cdot \chi_{JS}(T_i \parallel O_i)$$

depending on which divergence is applied, where m is the number of end users, p_i denotes the probability all possible traces of user u_i requesting a service s, T_i and O_i denote the random variables of actual traces and observed traces under condition of user u_i defined by $\mu_{\mathcal{T}}$ and $\mu_{\mathcal{O}}$, $\chi_{KL}(T_i \parallel O_i)$ and $\chi_{JS}(T_i \parallel O_i)$ denotes the distance-based information loss measure of user u_i.

3.7 An Application to Service Partitioning

In this part, a simple application of the quantitative analysis of opacity approach to service partitioning in a hybrid computing system is given. The entropy-based measurement and channel capacity computation can be used to help protecting clients' data in the hybrid computing environment. Specifically, given a type of observers of interest, based on the measurement achieved, services/data can be allocated into systems with different security level as required.

The entropy-based opacity measurement implies the degree of transparency of the communication channel between the clients and the services in the system under different security preserving mechanisms to the observers. For instance,

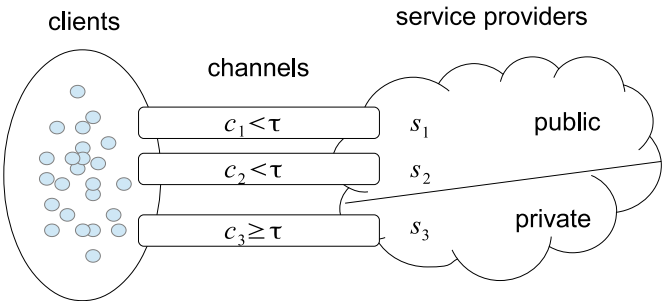


Figure 3.7: Application to service partitioning for hybrid system

we can move services which connect to lower capacity channels to public system, and move services which connect to higher capacity channels to private system. In particular, given a threshold τ, let the channel capacity regarding to a service S_i be C_i. If $C_i < \tau$, one can move service S_i to the public system; otherwise, one can move service S_i to private system.

For instance, Figure 3.7 shows the intuition of the basic idea discussed above. Assume we have a set of clients who are interacting with the service providers (S_1, S_2, S_3) located in the system. By applying the measurement mechanism, we get the capacity of channels to service S_1, S_2, S_3 denoted by C_1, C_2, C_3. Assume $C_1 < \tau$, $C_2 < \tau$, and $C_3 \geq \tau$, where τ is a given threshold. According to the basic rules, we can move S_1, S_2 to public site and S_3 to private site.

3.8 Summary

In this chapter, security lattice is introduced for cloud computing systems, and a formal model to provide flow-sensitive analysis of federated cloud computing system is presented. The entities located in the cloud computing systems are represented as a lattice. Opacity is introduced for cloud computing systems. A service-based computing system is defined based on the behaviour of users. Probabilistic and information theoretic means and equivalence relation are used to quantify the opacity in the system. This study can help service providers allocate services or resources into federated computing system.

Chapter 4

Cost-benefit Analysis of Enterprise Information Security Technology

Many organizations maintain sensitive information or documents that should be accessed only by authorized personnel, for example, personal health records in health institutions, bank statements and account balances for financial organizations. Confidential information leakage and sensitive information distortion have been identified as one of the major information security threats that cause reputation damage, identity theft and even threaten the viability of the company [27]. It is essential that companies and organizations keep these information and documents safe. Enterprise information security technologies (e.g., usb, DRM) are developed to address these concerns, use encryption to restrict the access of protected document to authorized end users.

The investment into an enterprise information security product involves massive uncertainty. Therefore, one major hurdle for implementing these products maybe that organization, as potential users, do not have an established procedure to evaluate the benefit, effectiveness, impact and cost of these products quantitatively.

Information security research has been traditionally focused on the technology and products themselves, for example, the architectures of the system, access

control, the functionality of the products. The impact of these products on business process and the impact of human behaviour on the effectiveness of the system have not been documented to date, although human behaviour has been identified as one of the critical factors that determine the effectiveness of security measures [1; 2].

The primary objective of this study is, therefore, to propose a standard procedure for organizations to conduct cost-benefit analyses for enterprise information security products using stochastic models. This procedure applies the concept of security metrics and Petri nets stochastic modelling to simulate the business process and human behaviour involved in enterprise information security technology. Potential cost and benefit of deploying enterprise information security products can be evaluated in quantitative terms. Using this procedure, not only different information security products can be compared in the same term, enterprise information security projects can also be compared with other projects in the organizations' portfolio in monetary terms. This information provides basis for organizations to make sound investment decisions.

4.1 Quantitative Evaluation Procedure (QEP)

The proposed evaluation procedure will focus on the following three aspects: the value or benefit that enterprise information security products can bring to the organization; the effectiveness of the products and factors that will impact the effectiveness; and finally, the cost and impact that information security products have on organizations.

In this procedure, information is treated as a business asset with varying levels of commercial values as introduced by Huebner and Britt [28] and Humphreys [29]. Since it is usually difficult to measure the benefits of information security products directly, costs of document disclosure or modification is measured instead to quantify the benefits as suggested by [30].

In addition, human behaviour is an integral part in the proposed procedure. Stochastic Petri nets models are used to simulate the behaviour and interactions of people who are involved in organizational functions. Human behaviour has been identified as one of the critical factors that determine the effectiveness of

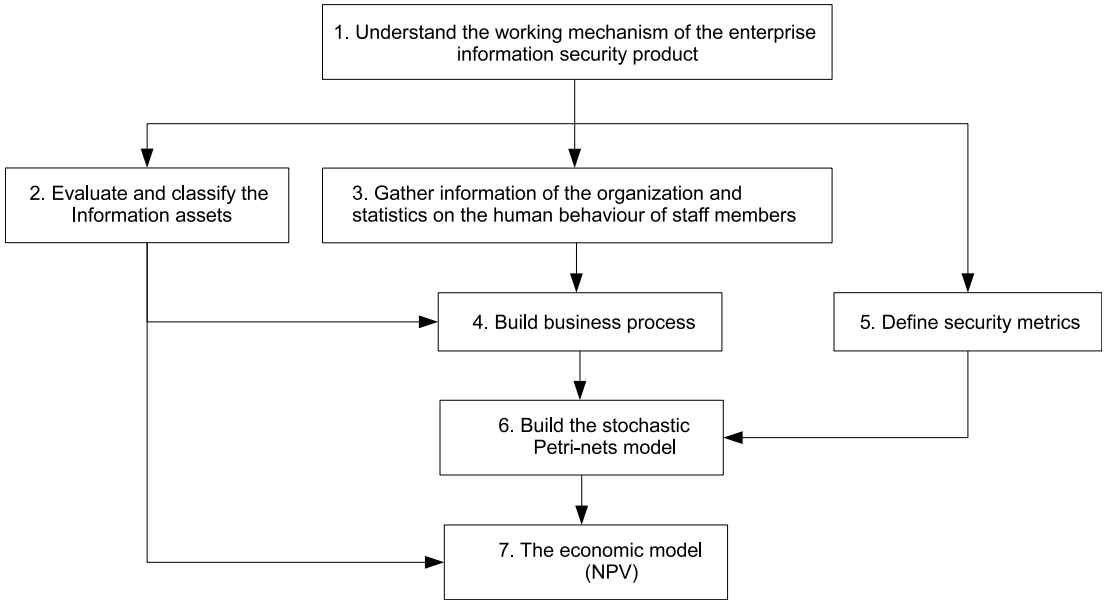


Figure 4.1: The flow chart of the proposed quantitative evaluation procedure (QEP) for enterprise information security products.

security measures [1; 2; 31]. For example, information security mechanism will not be effective, if an individual employee does not comply with it or is not aware of it [2]. Company policies, e.g. the way documents/information are classified, affect the effectiveness of the information products as well.

Finally, in the proposed procedure, a method of quantitatively evaluating cost and impact of the information security products in monetary terms is proposed. Some costs associated with these products are tangible, for example, the capital expenditures on the products and associated hardware/software and daily operational expenditures associated with maintenance. These tangible costs are easy to account for in monetary terms. There are also some intangible costs. For example, it has been documented that implementing a strict security mechanism will reduce the efficiency of the organization [2; 32]. The requirement of security behaviour often has conflicts, and competes for working hours with employee's production tasks [2]. In addition, encryption - a major information security function often reduces the availability of data and brings inconvenience to end users to use protected documents [32]. The impact of information security products on staff productivity is quantified in terms of non-productive time (NPT).

The proposed procedure for the quantitative evaluation of enterprise information security products is summarized in Figure 4.1 and described step by step below:

Step 1: Understand the working mechanism of the enterprise information security product in evaluation, by testing the products and by reading manuals or white papers provided by the product provider. Plot flow charts that represent the communications among users, servers, and administrators. These flow charts are the prototype of the business process. General information on the architecture, major components and major functionalities of the system can be available in previous publications.

Step 2: Evaluate and classify the organization's digital information assets. Study the organization's document classification policies, and classify the documents/information into different categories by their level of confidentiality according to company policies. Then assign estimated values to documents of each category. This information valuation process should be based on the business needs of the organization [29]. These values will be used in Steps 4 and 7 of the proposed procedure. Methods and processes to evaluate the value of documents can be found in published works, (e.g.,[29; 33; 34; 35]).

Step 3: Gather information about the organization and statistics on the human behaviour of staff members in the organization. These statistics include but not limited to: the total number of employees in the organization, the time that a user has to wait to receive assistance from an administrator etc. It has been reported that most security incidents are caused by human errors instead of technology failures [36], although security incidents can result from natural disasters, technical issues and human acts [37]. Human acts can be further classified into two categories: malicious or non-malicious. Malicious acts often include different types of human malicious acts [2]. Non-malicious acts are usually caused by users who did not understand or ignored security policies. Typical violations include sharing of passwords and not closing documents after viewing them, and so on [37].

Three main techniques have been established and widely used to collect and analyse human behaviour data: questionnaires, interviews, and action research. A questionnaire consists of a series of questions; respondents answer questions by

completing the questionnaire themselves [38]. An interview is a conversation between the interviewer and the interviewee, in which interviewer elicit information from the interviewee [38]. Action research is a reflective process of progressive problem solving. Researcher themselves are actively involved in the topic being researched. They work with each other to improve the current situation [39]. Baskerville [40] and Baskerville and Wood-Harper [41] introduced action research methods into information systems research. Stephanou [42] successfully applied the action research method to evaluate the impact of information security awareness training on users' information security behaviour.

Human behaviour data are critical inputs into the stochastic Petri nets model in Step 6. It might take significant time and effort to gather this information. However, the better the quality of the statistics on user behaviour, the more accurate the results of the evaluation of the informations security products will be. In general, if the model is more complex, more data are needed.

Step 4: Build a business process that includes a set of ordered activities to be undertaken by humans or other resources of the organization. This business process is a set of logically related tasks performed to achieve a defined business goal [31]. It is a structure for actions and implies on how work is done within an organization. These actions are work activities across time and space, with a beginning and an ending [43]. Users in the organization use documents to do their daily work. Document/information classification policies and human behaviour shapes the business process in the organization. Main business process modelling techniques can be found in various published research documents. (e.g., [44]).

Step 5: Define security metrics. Security metrics are a series of criteria that are defined to help organization measure the success of security investments; security metrics enable organization to know how well the security products or security strategies are meeting the security objectives. Most security metrics cover these four different aspects: perimeter defense capability, coverage and control, availability and reliability, application risks [45]. The value of each item in the metrics will be generated by the stochastic model in Step 6. For example, the number of documents protected by the digital rights management (DRM) product is a measure of DRM product perimeter defense capability; the number of workstations covered by the DRM product is measure of the coverage of control

of DRM product; the amount/percentage of documents that cannot be accessed by authorized users is a measure of the availability and reliability of the DRM product. The number of users who share password to open protected document is measure of the application risks of the DRM product. Information on choosing and defining security metrics can be found in previous publications (e.g., [45; 46]).

Step 6 : Construct and run the stochastic Petri nets model. In this step, the business process established in Step 4 is mapped into Petri nets application software. After the business process is mapped, parameters collected and defined in Step 3 are input into the model. Model outputs are used to calculate the security metrics that are defined in Step 5.

Step 7 : Account for all the tangible and intangible costs of the enterprise information security products and calculate the Net Present Value (NPV) of the implementation project using the results from Step 6. The calculated NPV is then used to help information security manager to make security investment decisions. Further information on NPV and economic model can be found in previous publications (e.g., [30; 47]).

4.2 Case Study

Digital Right Management (DRM) is one of many information security products, and DRM claims to have the ability to protect the confidentiality, integrity and availability of information in the organization [48; 49; 50; 51]. In this section, MS IRM (Microsoft Information Rights Management) is studied in order to illustrate the procedure proposed. MS IRM is a DRM product developed by Microsoft Corporation. It is a Microsoft Office component, and can be implemented through the Rights Management Services installed on Windows Server 2003 or 2008.

4.2.1 MS IRM System Analysis

According to the proposed procedure, the first step is to understand the working mechanism of the DRM by reading the DRM user manual and by testing using the software.

One key feature of the MS IRM is that it centrally manages documents and

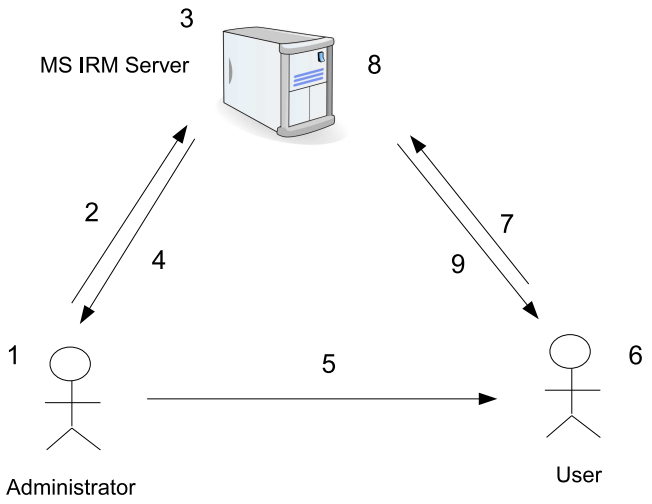


Figure 4.2: A simple representation of the working mechanism of MS IRM product. This chart illustrates the relationship between the MS IRM server, administrators, and users.

uses encryption to keep the information secure no matter where it has been transferred. Authorized users use decryption keys to open the document and access the information [52; 53]. Administrators who have unlimited access to this centralized space play a key role in the IRM business process. Administrators manage protected documents by defining and changing policies for all the documents and authorized users regardless of the document location. Administrators create new authorized users or delete existing authorized users from the user list of the document [52; 53]. The workflow of creating and viewing a protected document by authorized users is illustrated in Figure 4.2. When a user tries to open a protected document (process 7), the centralized server will check the authentication of the user. If the authentication fails, the user has to seek help from the administrator. The administrator will then check the identity of the user to make sure that the user has the status to access the document and add this user to the authorized user list of the particular document (processes 4 and 5).

The working mechanism in an MS IRM environment is described below:

1. An administrator uses an IRM-enabled application to create a normal document. When the administrator chooses the restrict permission option, the IRM controller is triggered. The IRM controller immediately requests a

username and a password from the administrator. When the administrator's response is sent to the server, the server validates the client. If the information is validated, the administrator can define users and users' rights for this document.

2. The IRM controller encrypts the document with a random symmetric key, and then sends a request for a publishing license directly to the IRM server. The request includes the random symmetric key, usage policies assigned to the encrypted document.

3. The IRM server encrypts the symmetric key with the server public key. Then the IRM server generates a publishing license, which includes the encrypted symmetric key, rights policy assigned to the encrypted document and the Uniform Resource Locator (URL) of the IRM server. IRM server then encrypts the publishing license with public key. Users and documents, rights and private key are kept in the IRM server.

4. The server returns the publishing license to the administrator side, and binds the publishing license to the encrypted document file.

5. Administrator distributes the document.

6. When a user tries to open a MS IRM protected document, the IRM controller is triggered on the user side, which requests the username and password information from this user.

7. The IRM controller sends a license request to IRM server. The license request includes the user's Rights Management user Account certificate (RAC), content identity, public key, the publishing license, and the rights policy information.

8. The license generator on the server checks whether or not the user is authorized (the user's identity is present in the user identities repository). If the user's identity does not match the present identity in the identities repository, the server rejects creating license. After the license generator confirms all the information, the license generator decrypts the symmetric

key using the private key of the server, re-encrypts it using the public key of the user, and creates user license (which contains the symmetric key for decrypting the document and the rights information) for the user.

9. User license is sent back to client side, the IRM controller uses license to decrypt and open the encrypted document.

4.2.2 Document Classification and Value

Sensitive documents can be classified into many levels according to their confidentiality and the impact in the event their confidentiality is compromised [29]. In this case study, sensitive documents in the organization are divided into two levels: confidential and strictly confidential documents. Users only need a username and password to open confidential documents; on the contrary, users need to have the particular password of each strictly confidential document in addition to user's own username and password. In this case study, it is assumed that there are 4000 sensitive documents in the organization that need to be protected by MS IRM. 50% of sensitive documents in the organization are classified as strictly confidential documents.

One reason for document classification is practicality. It is often time-consuming and cost prohibitive to assign a value to each single document that needs to be protected. Once the document is classified, however, a single value can be assigned to a particular class. In this case study, the financial value of a confidential document is assigned to be $£10,000$ and the value of a strictly confidential document is assigned to be $£25,000$ in confidentiality terms to the organization [29].

4.2.3 User Behaviour Study

In this study, a set of assumptions about user behaviours are made (Table 4.1), which can be used to show the methodology we proposed. In a real business case, user behaviour data should be collected instead of assumed. In the next chapter, we will provide more information about the parameters.

Table 4.1: Data on human behaviour within the organization. Human behaviour is an important factor in the effectiveness of DRM products. User behaviour data are critical input parameters of the stochastic Petri nets model. In general, they are gathered using questionnaire, interview, or action research methods.

Parameters	**Value**
Number of documents, an user might use every day on average	4
Number of normal working hours per day	8
Number of working weeks per year	40
Number of administrators in the organization	2
The percentage of working time, unauthorized users get opportunities to read documents	0.025%
The percentage of time when authorized users experience a login system failure	1%
The percentage of time, users cannot remember the password for strictly confidential documents	2%
Percentage of unauthorized users use username/password to login in successfully	20%
Percentage of unauthorized users use document password to login in successfully	20%
The average time users need to spend to pass user authentication	15 seconds
The average time users need to spend to pass document authentication	15 seconds
The average time administrators need to help each user	10 minutes
The average time authorized users can wait for help	2 hours

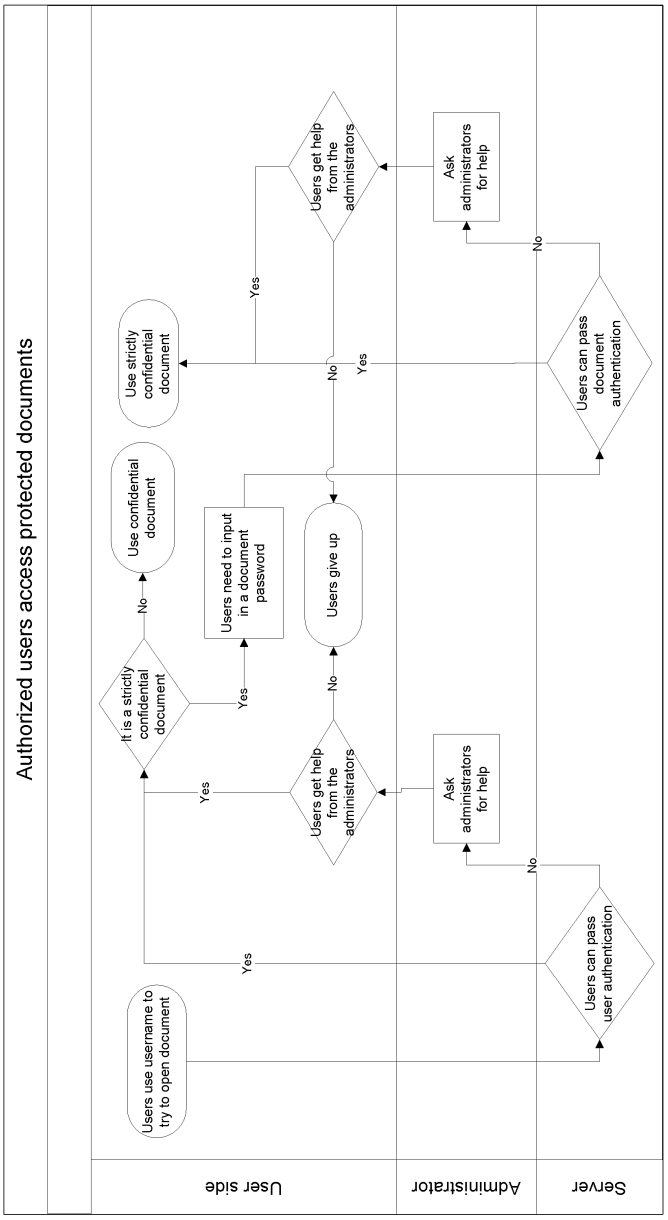


Figure 4.3: The flow chart illustrating business process of authorized user's access to documents under MS IRM protection. The chart is another and more detailed representation of the relationship between the MS IRM server, administrators and users. Authorized users need to pass authentication in order to access protected documents. In case they cannot pass authentication, they need help from administrators to gain access.

4.2.4 Business Process of MS IRM

One key step in the proposed procedure is to describe the business process associated with the working mechanism of the DRM product. The business process of using MS IRM system to protect confidential document is illustrated in a flow chart that outlines the interactions between authorized users, system administrators and the MS IRM system (Figure 4.3).

Potential users of a document can be classified into two groups, authorized users and unauthorized users. For confidential document, MS IRM requires users to log in using user's log-in ID and password; for strictly confidential documents, a document specific password is needed additionally. Authorized users should possess correct passwords to open documents for which they have rights. However, authorized users might forget the password, especially those of the document specific passwords for strictly confidential documents, because they might not open these documents frequently. In this case, the user will have to contact the administrator to retrieve the password. Administrators, however, are not always available, because they could be still handling previous requests. This will reduce the efficiency of the organization and will result in non-productive time, NPT [54].

For unauthorized users, since they are not supposed to possess the correct password, the chance for them to open a confidential document is reduced; for strict confidential document, the chance is even smaller (Figure 4.4). However, there are still cases that unauthorized users can open protected documents. Unauthorized users might try a certain number of times and open the document successfully; or they might have acquired the correct passwords because of security leaks etc.

4.2.5 Security Metrics

Step 5 of the proposed procedure is to define security metrics in order to measure the confidentiality and availability of documents. MS IRM uses encryption to keep documents secure; this process decreases the availability of documents to unauthorized users. As discussed previously, the value of the implementation of MS IRM product is best measured by the cumulative value of all the confidential

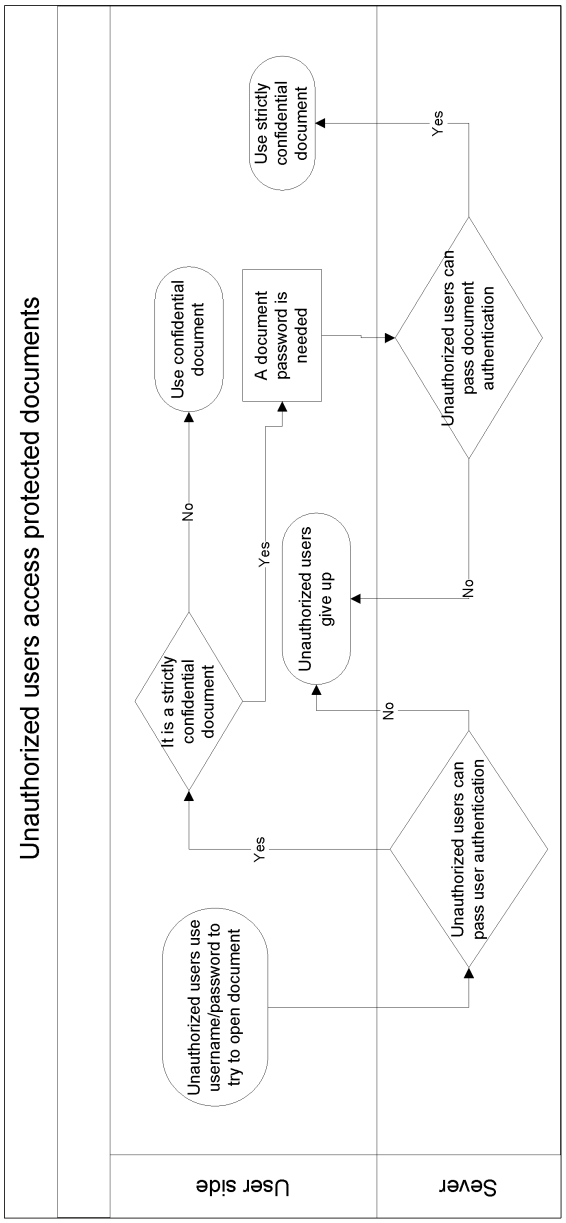


Figure 4.4: The flow chart illustrating business process of unauthorized user's access to documents under MS IRM protection. In this case, if unauthorized users try to open protected document, they will not receive assistance from the administrators when they cannot pass authentication.

Table 4.2: Security metrics defined to evaluate the effectiveness of the MS IRM system. It is in general difficult to measure the benefit of DRM product. The potential cost when document is leaked or compromised is usually defined instead. The number of confidential and strictly confidential documents is therefore defined to evaluate the benefit of the MS IRM products.

	User's Type	Executive	Document Type
1	unauthorized	cannot read	strictly confidential
2	unauthorized	cannot read	confidential

documents if they were disclosed to or modified by unauthorized users. The metrics for MS IRM are defined in Table 4.2.

4.2.6 Stochastic Models Description and Results

Step 6 of the proposed procedure is to map the business into a stochastic Petri net model and run this model. Data collected in Steps 2, 3 and 4 are input into the Petri nets model. The value of security metrics defined in Step 5 and the non-productive time will be generated by the simulation runs.

Figure 4.5 is a Petri nets model that was generated from the business process model (Figures 4.3 and 4.4). The model consists of 14 places, 15 transitions (6 timed transitions and 9 immediate transitions), 3 input gates and 3 output gates. The input gates and output gates control the enabling of activities and define the making changes that will occur when activities complete.

Users of the documents in the organization (*Users*) can be grouped into two categories, authorized and unauthorized users, therefore, the transaction *Doc_use* have two branches and can take the token to two places, *Authorized_User* or *Unauthorized_User*, respectively.

For authorized users, the main model objective is to quantify the reduction of working efficiency because of the changed workflow after installing MS IRM. As discussed previously, the NPT associated with the implementation of DRM products is one of the main negative impacts on the organizations. In this proposed procedure, the NPT is modelled by timed transitions of stochastic Petri nets. Two types of delays can be incurred by the implementation of MS IRM,

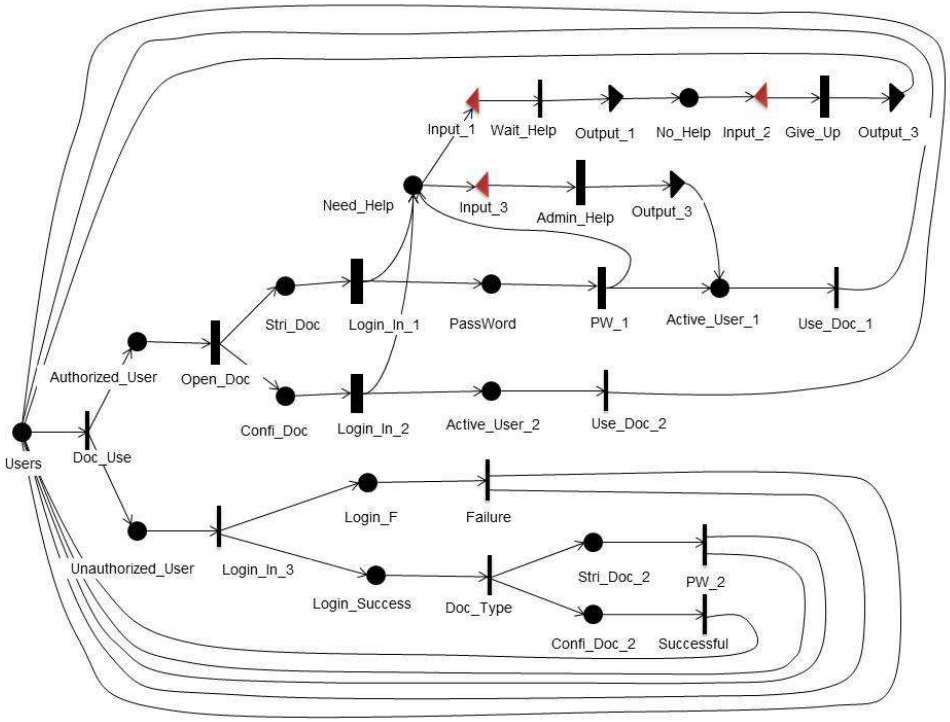


Figure 4.5: A Petri nets model generated by mapping the business process associated with both authorized users and unauthorized users into the Möbius software. A Petri nets model consists of places, transitions, and arcs that connect them. In this figure, circles represent places, vertical lines represent transitions, lines represent arcs, and triangles represent input gates and output gates.

the delay to pass authentication and the delay to wait for administrators' help.

When authorized users try to open documents, whether the document is a confidential document (*Confi_Doc*) or strictly confidential document (*Stri_Doc*), the user needs to have a username and password in order to login to the system. These authentication processes take a certain amount of time, and is accounted as *Login_In_1*. For strictly confidential document, the user further needs to input in a document password (*PassWord*), which takes another certain amount of time, given by the time taken by *PW_1*. If the user passes the document authentication, then the user can open the document and become an active user (*Active_User_1*). However, if the user cannot pass the document authentication or the user cannot pass the user authentication, the user will have to contact the administrators (*Need_Help*) for help. The time delay that the user has to wait for

the administrators is accounted as *Admin_Help*.

For unauthorized users, the main model objective is to quantify the reduction in login-in success rate, after the organization implements the MS IRM system. Unauthorized users will first have to log in to access any document, which is represented by the transaction *login_In_3*. This process is modeled as a token starting in the place of *Unauthorized_User*. If the login fails, the token will move to the place, *Login_F*. Then the transaction *Failure* will lead the token back to the original place, *Users*. If the unauthorized user passed the first authentication process of *Log_In_3*, the token will move to the place, *Login_Success*. Since there are two types of documents defined in the organization, when the unauthorized user tries to open a document, the transaction *Doc_Type*, can lead the token into two places, *Stri_Doc_2* (strictly confidential document) or *Confi_Doc_2* (regular confidential document). If the token goes to the place of *Confi_Doc_2*, the unauthorized user has access to the document. This is represented by the transaction Successful that leads the token back to the original place, *Users*. If the token goes to the place of *Stri_Doc_2*, another authentication process will take place. This is represented by the transaction, *PW_2*. *PW_2* has two branches, successful and unsuccessful, both of which lead the token back to the place *Users*.

Two security metrics are defined previously to measure the effectiveness of the MS IRM system in Step 5 (Table 4.2), the number of times that unauthorized users cannot read strictly confidential documents and confidential documents. The values of security metrics are represented by throughput of transactions when the model is run 96000 time units. The number of tokens that move through the transition, *PW_2 (case2)* and *Failure (case1)*, represents the number of strictly confidential documents unauthorized users cannot read. The number of token that moves through the transaction, *Failure (case2)*, is the number of confidential documents unauthorized users cannot read.

The behaviour of the model can be measured by the *Impulse Rewards* model and *Rate Rewards* model in the *Möbius* software [55]. The throughput of transitions is computed according to the formula which is described in Section 2.3.2:

$$\sum_{a \in A} \mathfrak{C}_a N^a_{[t,t+1]}$$

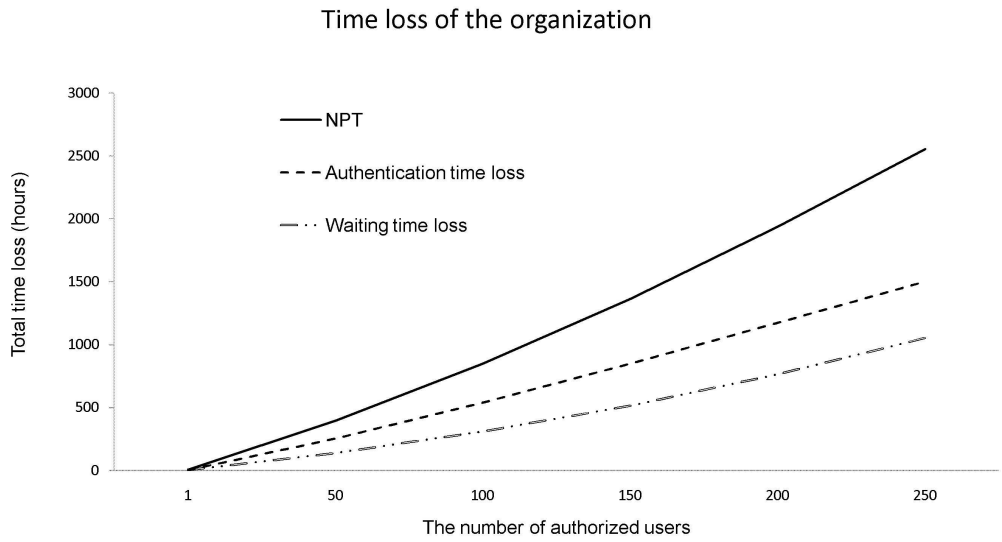


Figure 4.6: Total non-productive time (NPT) associated with the deployment of MS IRM system. NPT increases significantly when the number of authorized users served by each administrator increases. The total NPT is composed of two components: the time loss associated with authentication process and the time loss associated with waiting on administrators when authorized users fail to pass the authentication process.

The number of tokens in sets of places are computed according to the formula:

$$\sum_{v \in \mathcal{P}(P,\mathcal{N})} \mathcal{R}_{(v)} M^{v}_{[t,t+1]}$$

From the result of this study, the deployment of the DRM product has a significant impact on the operational efficiency of the organization. Within one year of the deployment of MS IRM in the network system in this case study (96000 time units in the model), a middle-sized organization that has 50 authorized users will incur about 396 hours, or about 49.5 days, of non-productive time (Figure 4.6), under the assumptions made in Table 4.1. This total loss of productive time has two components: the time spent on authentication procedures, around 255 hours and the time spent on waiting for responses from the administrators, around 141 hours (Figure 4.6).

Under the assumption made in this study in Section 4.2.3 (Table 4.1), there are 10 security attacks every year, in which unauthorized users attempt to read

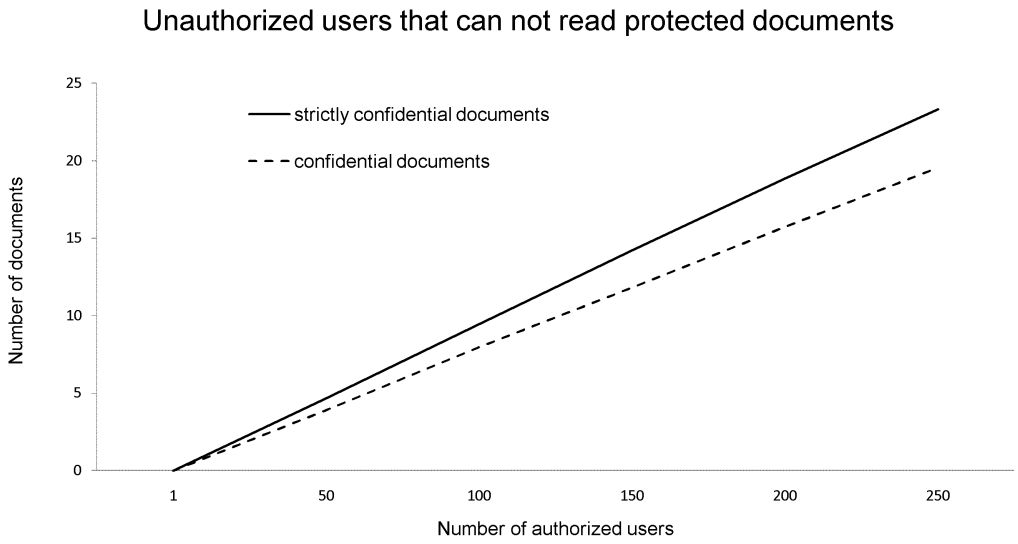


Figure 4.7: Total number of documents that unauthorized users cannot read associated with the deployment of MS IRM under the assumptions made in Table 4.1.

sensitive document. According to simulation results, as a result of MS IRM deployment, around 5 strictly confidential documents are prevented from reading by unauthorized users, and 4 confidential documents are prevented from reading by unauthorized users (Figure 4.7).

4.2.7 Economic Models

In this section, the benefits and costs of implementing MS IRM products are compared using the concept of NPV [37].

As discussed previously, the benefit of the MS IRM system is the reduction in access of unauthorized users to confidential documents. In this case study, after using the MS IRM system for one year, 5 strictly confidential documents are prevented from reading by unauthorized users and 4 confidential documents are prevented from reading by unauthorized users (Figure 4.7). Using the assumed document value in Section 4.2.2, the annual benefits on documents disclosure amounts to £165,000 (Table 4.3).

The capital expenditure of implementing MS IRM system includes the purchase of MS IRM system itself and any associated costs to upgrade hardware and

Table 4.3: Benefits analysis of the MS IRM product. The benefit of the MS IRM system includes a reduction in unauthorized access to confidential documents.

Annual benefits analysis on the confidentiality of documents by using MS IRM			
	Number of documents	Single document value	Total value
Strictly confidential documents that are prevented from being read by unauthorized users	5	£25,000	£125,000
Confidential documents that are prevented from being read by unauthorized users	4	£10,000	£40,000
Total benefits:	£165,000		

software. In addition, each year, 396 hours of non-productive time incurred as the result of the reduction in organizational efficiency after the implementation of the MS IRM system (Figure 4.6). In order to translate NPT into monetary terms, the average salary of £60 per hour is used, therefore, in this study, the annual cost on NPT amounts to £23,760. For many projects, this is a conservative estimate, because of high daily operational rate, for example, the daily rates of drilling rigs in oil and gas industry. In addition, two administrators have to be hired to handle the MS IRM system and the requests in the organization. An administrator's annual salary is around £32,000 [56]; therefore, the administrators' costs are around £64,000 per year. Training of employees in this case is assumed to cost £20,000 per year. The total costs associated with implementing MS IRM system is listed in Table 4.4.

We put the benefits and costs (Table 4.3 and Table 4.4) associated with implementing MS IRM system into the NPV, the NPV is represented in Section 2.8. Therefore, the NPV of implementing the MS IRM can be calculated as following [37]:

NPV at year one ($T = 1$):

$$NPV(i,T) = NPV(0.5,1) = -K + \sum_{t=0}^{T} \frac{R_t - C_t}{(1+i)^t} = -K + \frac{R_1 - C_1}{(1+i)^1}$$

$$= -£21,121.65 + \frac{£165,000 - £107,760}{(1+0.05)^1}$$

Table 4.4: Costs analysis of the MS IRM product. The costs of MS IRM product include initial capital expenditure on MS IRM software and associated hardware and software upgrade cost. In addition, annual operational costs include administrator salaries, non-productive time associated with the reduction of operational efficiency and employee training costs.

Initial capital expenditure costs by deploying MS IRM	
Type of costs	Value
MS IRM software itself	£1,121.65
Upgrade hardware or operating system	£20,000
Total costs:	£21,121.65
Annual operational costs by implementing MS IRM	
Type of costs	Value
Employee non-productive loss	£23,760
Administrator costs	£64,000
Training employees	£20,000
Total costs:	£107,760

$= £33,392.64$

NPV at year two ($T = 2$):

$NPV(i,T) = NPV(0.5,2) = -K + \sum_{t=0}^{T} \frac{R_t - C_t}{(1+i)^t} = -K + \frac{R_1 - C_1}{(1+i)^1} + \frac{R_2 - C_2}{(1+i)^2}$

$= -£21,121.65 + \frac{£165,000 - £107,760}{(1+0.05)^1} + \frac{£165,000 - £107,760}{(1+0.05)^2}$

$= £85,311.01$

In which, R_t represents the benefit of year t; C_t represents the cost of year t; K is the initial capital expenditure. The discount rate i is assumed to be 0.05.

In this studied case, the NPV of implementing MS IRM is larger than 0. Therefore, the implementation project will bring positive cash flow to the organization. It is, therefore, recommended to implement MS IRM into the organization's network. However, as discussed previously, this is the optimal case, in which the capital of the organization is unconstrained. For an organization with limited capital that has many profitable projects, projects with positive NPV will be compared against each other and the ones that have higher rate of return might be chosen for investment.

4.3 Summary

To assist security investors to make sensible investment decisions, a procedure that can quantitatively evaluate the benefits and costs of implementing enterprise information security products was proposed. The deployment of MS IRM system was used as a case study. In this example, the mechanism of MS IRM system is analysed; flow charts that represent the communication between users and server, administrator and server are generated. Then human behaviour of staff members in the organization is studied. In addition, business process is built to represent the set of ordered activities to be undertaken by staff or other resources of the organization. A group of security metrics were developed to measure the effectiveness of the MS IRM system. Furthermore, the stochastic Petri Nets method is used to simulate and predict the impact of the deployment of the system on normal business processes. The simulation results provided important information that the business needed for making decisions on whether or not to implement this particular type of enterprise information security system.

This procedure was developed for the project in this particular case study under a set of assumptions; however, it has the potential to be used as a general practice during the procurement process of information security products. Different assumptions can be made and different parameters can be used based on the data collection of user behaviour study. These parameters are used in the stochastic model to tailor the needs and requirements of the security investor.

Chapter 5

Data Resources in Dynamic Environments

In chapter 4, we proposed a methodology using stochastic Petri nets for quantitative analysis of the effectiveness of the enterprise information security technologies, and we found that an important advantage of the implementation of an information security technology system is the reduction of unauthorized attempts to access data resources. However, the previous studies did not consider the mobility of the users, and the level of exposure to threats of data resources in different locations.

In this chapter, we will analyse data resources in a dynamic environment, and we will focus on the location of data resources and the location of users, as well as security policies in an organization. This study can help the technical experts to provide a deeper analysis of the detailed security measures required, and help the decision makers to know the efficiency of a given information security technology.

5.1 Data Resources in an Organization

Figure 5.1 illustrates the way in which users handle the data owned by an organization. They store sensitive data on their desktop PCs, smart phones or USB sticks. When such a device is lost, the organization is unable to exert any control on the information stored on this device. In addition, users can occasionally send

email messages that contain confidential files as attachments without noticing that the files should not be distributed, or that the recipient should not see the files, or that an unintended recipient has been incorrectly selected to receive the email. Each of these behaviours can cause the digital information to leak outside the company.

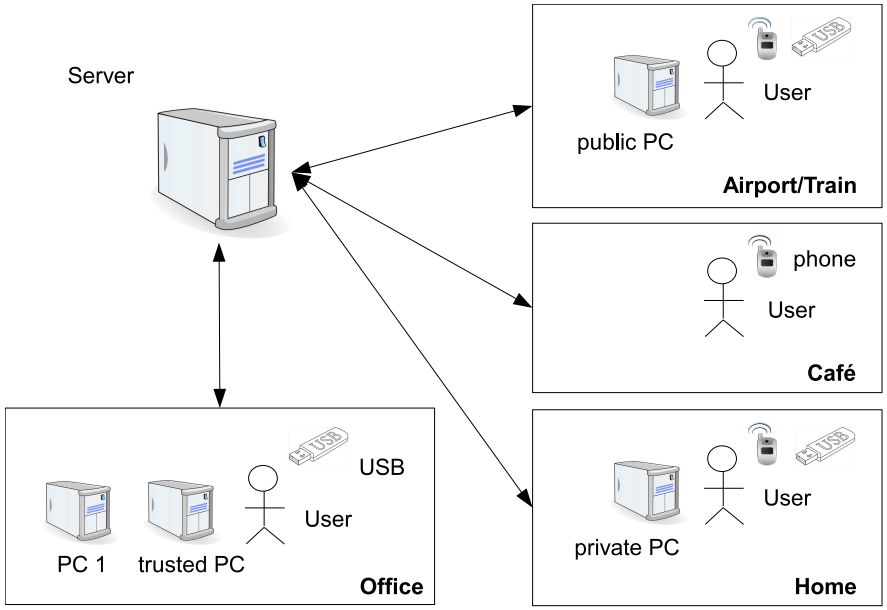


Figure 5.1: Users access data through different devices and in different locations.

Information security technologies[1] have been developed to address these concerns. In [57], the authors survey the existing enterprise technologies that control access to confidential digital data (e.g., USB access control solutions, digital rights management software, and disk encryption techniques). The researched technologies use endpoint access control as a means of limiting the maintenance overhead introduced by unauthorized devices. They also provide auditing options and prevent outsider access through encryption, which can reduce the loss or leaking of data. It has been found that the various information security measures are reliant on the cooperation of various people and system components, and it has been identified that the effect of the information technologies highly depends on the user behaviour.

[1]We refer to these information security technologies as access control solutions or identity and access management technologies.

5.1.1 Threats

In business processes there are various types of threats that could occur and lead to the loss or leaking of data. Different threats have different probability of occurring; in particular, there are two important factors in this respect: *accidental loss*, and *information theft targeted by criminals*. An accidental loss may be somewhat quantifiable, particularly as it is related to physical devices such as laptops or smart phones. The targeting of information by criminals will depend both on the individuals with the capability to access data and their capacity for criminal behaviour, and an opportunity to perform the crime. Factors influencing both accidental loss and crime will be changeable over time and hence it becomes important to consider security related decisions within a wide range of threat contexts. Information resources are subject to many kinds of threat. Here we first need to define the threats. Note that, in general, a threat is a natural disaster, an unintentional act by an individual that causes harm, or a malicious act by an individual or group of individuals [58; 59].

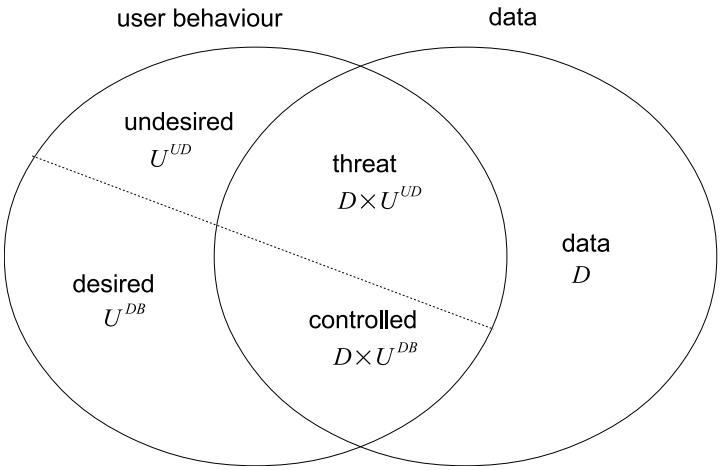


Figure 5.2: User behaviour impacts on the security of data resources.

Figure 5.2 shows the conditions causing threats which result from the interaction between data resources and users behaviour. Here, and later in the paper, it is assumed that D is the set of data resources in an organization, and d is a specific element in D. U is the set of all users whose behaviours could be applied to D, and u is a specific element in U. The user behaviour fall into two cate-

gories, namely users with *desired behaviour*, $u^{db} \in U^{DB}$, and users with *undesired behaviour*, $u^{ub} \in U^{UB}$, i.e.,

$$U = U^{DB} \uplus U^{UB} .$$

The set of threats, including unintentional acts by users that cause harm (e.g., device loss and miss-sent messages), is denoted by TS, and ts is a specific threat in TS. In Figure 5.2, a threat results from an undesired user behaviour applied to a data resource. Therefore, we have:

$$TS = D \times U^{UD} .$$

Moreover, a controlled data resource results from a desired user behaviour applied to a data resource. Therefore, we have:

$$CD = D \times U^{DB} .$$

In what follows, we will refine the definition of TS and CD, after taking into account the impact of potential data protection measures adopted by an organization.

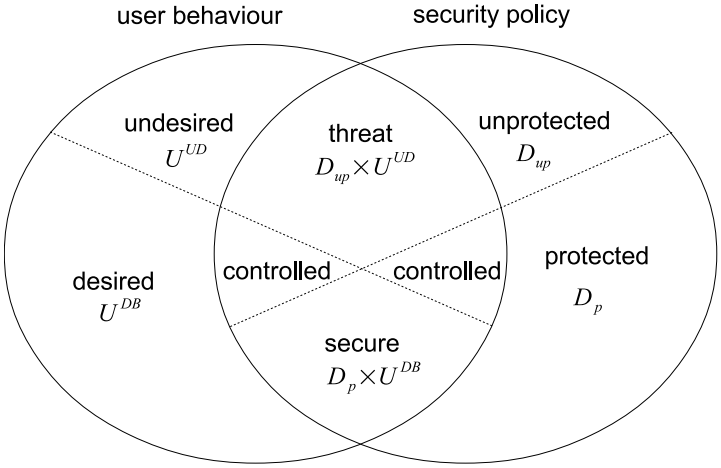


Figure 5.3: Security policy impacts on the security of data resources. The conditions causing threats result from the combination of unprotected data resources and undesired user behaviour.

5.1.2 Data Resources Protection Polices

A data resources protection domain consists of all the elements of an organization that are subject to the same data resources protection policy. One can identify various information security technologies preventing outsider access to data resources through access control [57]. Therefore, we consider that there are two categories of data resulting from the implementation of information security technologies: *protected data*, $d_p \in D_p$, and *unprotected data*, $d_{up} \in D_{up}$, i.e.,

$$D = D_p \uplus D_{up} \ .$$

The interplay of a data protection policy implemented by an organization with the behaviour of the users is illustrated in Figure 5.3. It shows some regions resulting from the combination of the protected and unprotected data resources with the desired and undesired user behaviours. The regions correspond to the situations listed in Table 5.1. Looking at Figure 5.3, it is clear that by minimizing the region resulting from an interaction between unprotected data resources and undesired user behaviours, one can reduce the level of security threats in the system.

Table 5.1: The interplay of security policy with the behaviour of users

secure	$D_p \times U^{DB}$	*protected data & desired behaviour*
controlled$_1$	$D_{up} \times U^{DB}$	*unprotected data & desired behaviour*
controlled$_2$	$D_p \times U^{UD}$	*protected data & undesired behaviour*
threat	$D_{up} \times U^{UD}$	*unprotected data & undesired behaviour*

After taking into account the implemented information security technologies protecting data resources, we can introduce *secure data* SD as a combination of protected data and users with desired behaviour, and then refine the threat model presented in Section 5.1.1, in the following way:

$$\begin{aligned} TS_{drp} &= D_{up} \times U^{UD} \\ SD_{drp} &= D_p \times U^{DB} \\ CD_{drp} &= (D_{up} \times U^{DB}) \uplus (D_p \times U^{UD}) \end{aligned} \tag{5.1}$$

5.1.3 Threat Environment

A threat environment generates threat events that may cause, e.g., a data loss incident. There are several potential threats that could occur and lead to security incidents involving data, each of which would normally have a different probability of occurring in different environments. Crucially, the same incident may bring a different impact to the organization in a different location. For example, a laptop might be left unattended in a public place allowing passers-by to read confidential emails, whereas a laptop left unattended in a private space might not result in any damage to the organization.

To model different threat environments in the system, we will use a set Loc of *locations*, with $loc \in Loc$ being a specific location. After that the threat model defined in (5.1) can be further refined to take into account different environments, in the following way:

$$\begin{array}{rcl} TS_{env} & = & TS_{drp} \times Loc \\ SD_{env} & = & SD_{drp} \times Loc \\ CD_{env} & = & CD_{drp} \times Loc \ . \end{array} \tag{5.2}$$

Thus, for instance, $TS_{env} = D_{up} \times U^{UD} \times Loc$.

5.1.4 Status of the Data Resources

We consider that each threat can lead to an incident [58]. In the model of data resources we are developing, we will identify security incidents with all system events in which a user exhibiting an undesired behaviour accessed unprotected data. We will use a special set I to record such events. In addition, we will use special sets, SEC and CON, to respectively record events in which data resources were accessed in a secure and controlled manner. We therefore have:

$$\begin{array}{rcl} I & \subseteq & D \times Loc \\ SEC & \subseteq & D \times Loc \\ CON & \subseteq & D \times Loc \end{array} \tag{5.3}$$

where the set of incidents, secure data accesses, and controlled data accesses are represented together with the locations at which they occurred. Note that the above definition could be extended to include other relevant information, e.g., the security level of data resource, or by allowing I, SEC, CON to be multisets (as in the rest of this paper).

5.1.5 Information Lattices

A lattice for security concerns $\mathcal{L} = (L, \leq)$ consists of a set L and a partial order relation $\leq$ such that, for all $l, l' \in L$, there exists a least upper bound $l \oplus l' \in L$ and a greatest lower bound $l \otimes l' \in L$. The lattice is complete if each subset L' of L has both a least upper bound $\coprod L'$ and a greatest lower bound $\prod L'$ [5; 6].

We will assign a security (or confidentiality) level $l \in \mathcal{L}_{sec}$ to each data item which will in practice be related to the degree of security of its contents. Moreover, each each location $loc \in Loc$ has a security level $l(loc) \in \mathcal{L}_{sec}$, and $l(loc)$ specifies the highest allowed security level of the data resources located in loc.

In the next section, we will put together into a single definition the various notions and notations described so far separately.

5.2 Data Resources in Dynamic Environments

The Petri net in Figure 5.4 models a dynamic data resources system. There is one user and one data resource in location loc_1. The user can enter in location loc_2, and then move to location loc_3. The data, e.g., can enter in location loc_2, and then move back to loc_1. Note that the user and data are represented by 'black' tokens for simplicity. In fact, they are 'coloured' tokens with associated attributes, such as security levels.

To facilitate the discussion, the net of Figure 5.4 is decomposed into three parts: *static data resources sub-systems*, *data flow* and *user flow*. The static data resources sub-systems show the interaction between users and data at individual locations, and the data/user flow sub-systems show the movement of the data/users between different locations.

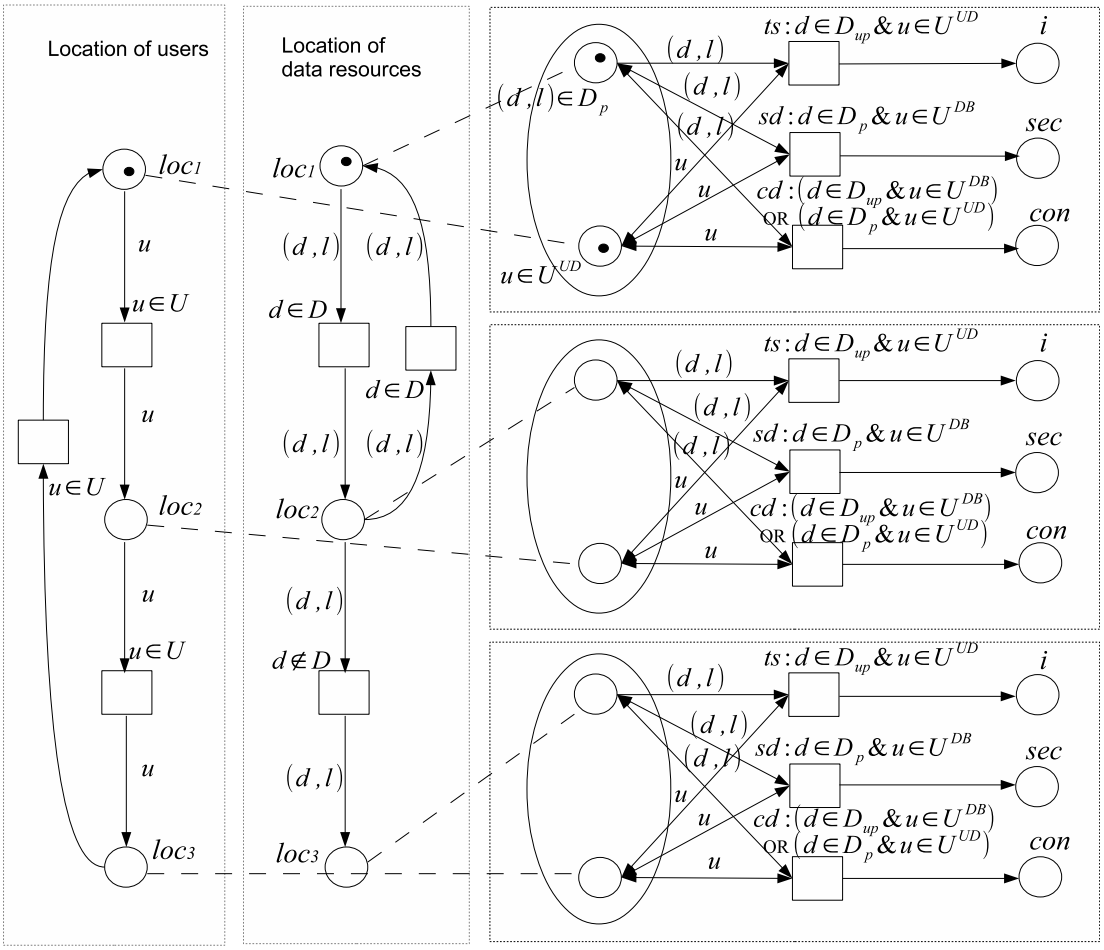


Figure 5.4: Data resources in a dynamic environment. Note that the dashed lines indicate that the joined circles represent the same place.

We are now ready to introduce a formal model of data resources in a dynamic environment.

Definition 32 (ddrs)**.** *A* dynamic data resource system *for security analysis is a tuple:*

$$DDRS = (D, U, \mathcal{L}_{sec}, Loc, \mathcal{A}, s_{init}) \tag{5.4}$$

where:

- $D = D_p \uplus D_{up}$ *is a finite non-empty set of* data resources*;*

- $U = U^{DB} \uplus U^{UD}$ *is a finite non-empty set of* users*;*

- $\mathcal{L}_{sec} = (L, \leq)$ *is a complete security lattice;*

- *Loc is a finite non-empty set of* locations *such that each location* $loc \in Loc$ *has a security level* l_{loc}*;*

- $\mathcal{A}$ *is a finite set of* actions *partitioned into three sets,* $\mathcal{A} = \mathcal{A}_{sdrs} \uplus \mathcal{A}_{df} \uplus \mathcal{A}_{uf}$*, each action being a pair*

$$\phi = (\phi^{in}, \phi^{out})$$

of finite multisets over the set

$$\begin{aligned} \mathcal{C} \;=\; & U \times Loc \cup \\ & D \times L \times Loc \cup \\ & \{i, sec, con\} \times D \times Loc\,; \end{aligned}$$

- s_{init} *is an* initial state *defined as a finite multiset over the set of tuples* $\mathcal{C}$*.*

In general, a state *of DDRS is a finite multiset over the set of tuples* $\mathcal{C}$*.* $\Diamond$

Note that the tags $\{i, sec, con\}$ are used to indicate the (multi)sets I, SEC, CON discussed in Section 5.1.4.

A data resource can have several copies, and each of these copies can have a different security level and reside in a different location. We further allow multiple copies of a single data to be present in a single location. As a result, a *state* is a multiset s over the set $\mathcal{C}$ rather than a subset of $\mathcal{C}$. Thus, for example, if $s(d, 2, loc) = 4$ then we know that, in the current state s, there are 4 copies of data d with security level 2 residing at location loc. We will also say that (d, l, loc) is present in state s if $s(d, l, loc) > 0$. If $s(u, loc) = 3$, then we interpret this as saying that there are 3 users with the behaviour u at location loc.

The decomposition of the Petri net of Figure 5.4 into three sub-models is reflected in the above definition by three types of actions: $\mathcal{A}_{sdrs}$ for the static data resources sub-systems, $\mathcal{A}_{df}$ for the data flow sub-system, and $\mathcal{A}_{uf}$ for the user flow sub-system. They will be presented in detail later on. First, we define how the system can execute actions, both one-by-one, and in groups of multisets.

Definition 33 (single action executions)**.** *An action* $\phi = (\phi^{in}, \phi^{out})$ *is* enabled *at state* s *if* $\phi^{in} \leq s$*, i.e., if the whole of* ϕ^{in} *is included in* s*. Such an action can*

then be executed *leading to a new state s' given by:*

$$s' = s - \phi^{in} + \phi^{out} \ ,$$

where $(-)$ and $(+)$ are multiset subtraction and addition, respectively.
We denote this by $s \xrightarrow{\phi} s'$. ◊

Note that the class of allowed action types may easily be extended to include, for example, checking for the absence of certain kinds of data.

Definition 34. *(multiset actions execution). A multiset of actions $\Phi = \{\phi_1, \ldots, \phi_n\}$ is* enabled *at state s if $\Phi^{in} \leq s$, where*

$$\Phi^{in} = \phi_1^{in} + \ldots + \phi_n^{in} \ .$$

Such an action multiset can then be executed *leading to a new state s' given by:*

$$s' = s - \Phi^{in} + \Phi^{out} \ ,$$

where

$$\Phi^{out} = \phi_1^{out} + \ldots + \phi_n^{out} \ .$$

We denote this by $s \xrightarrow{\Phi} s'$. ◊

With such a definition we can define precisely what are the states which can be reached from the initial one.

Definition 35. *(reachable states). The set of reachable states of the dynamic data resource system DDRS of Definition 32 is the minimal set of states RS containing the initial state s_{init} and such that if $s \in RS$ and $s \xrightarrow{\Phi} s'$, for some Φ, then $s' \in RS$.* ◊

The above framework is still not practical, as it allows too general a form of the actions. We will now address this by introducing specific forms of the actions in the sets $\mathcal{A}_{sdrs}$, $\mathcal{A}_{df}$ and $\mathcal{A}_{uf}$.

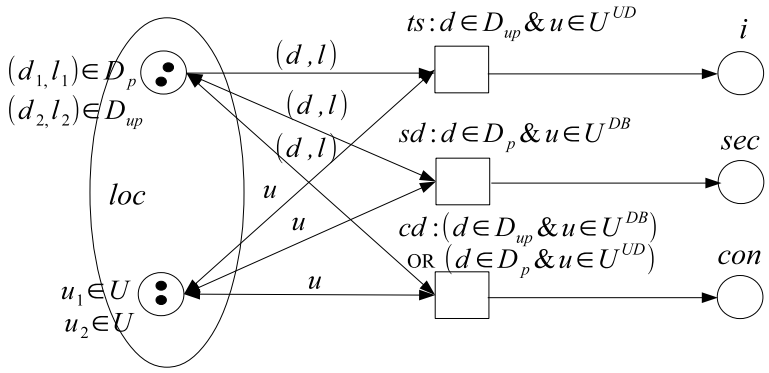


Figure 5.5: Basic structure of the data resources sub-system at location *loc*. There are two users, u_1 and u_2, and two data resources, d_1 and d_2.

5.2.1 Static Data Resources Sub-systems

Figure 5.5 shows the structure a data resources sub-system (a central component of our model), representing the behaviour of the users and data in a single location.

The set of actions $\mathcal{A}_{sdrs}$ is made up of four distinct kinds of actions:

$$\mathcal{A}_{sdrs} = \mathcal{A}_{sdrs}^{(i)} \uplus \mathcal{A}_{sdrs}^{(sec)} \uplus \mathcal{A}_{sdrs}^{(con)} \uplus \mathcal{A}_{sdrs}^{(data)} \ ,$$

defined in the following way (note that a term of the form $(x, \ldots, y)@loc$ denotes a tuple $(x, \ldots, y, loc)$):

- Each $\phi = (\phi^{in}, \phi^{out}) \in \mathcal{A}_{sdrs}^{(i)}$ is such that

$$\begin{aligned} \phi^{in} &= \{u@loc, (d, l)@loc\} \\ \phi^{out} &= \{u@loc, (i, d)@loc\} \end{aligned}$$

  where $u \in U^{UD}$, $d \in D_{up}$, $l \in L$ and $loc \in Loc$.

- Each $\phi = (\phi^{in}, \phi^{out}) \in \mathcal{A}_{sdrs}^{(sec)}$ is such that

$$\begin{aligned} \phi^{in} &= \{u@loc, (d, l)@loc\} \\ \phi^{out} &= \left\{ \begin{array}{l} u@loc, (sec, d)@loc, \\ (d_1, l_1)@loc, \ldots, (d_m, l_m)@loc \end{array} \right\} \end{aligned}$$

where $u \in U^{DB}$, $d \in D_p$, $l, l_1, \ldots, l_m \in L$, $d_1, \ldots, d_m \in D$, and $loc \in Loc$.

- Each action $\phi = (\phi^{in}, \phi^{out}) \in \mathcal{A}_{sdrs}^{(con)}$ is such that

$$\begin{aligned} \phi^{in} &= \{u@loc, (d,l)@loc\} \\ \phi^{out} &= \left\{ \begin{array}{l} u@loc, (con,d)@loc, \\ (d_1,l_1)@loc, \ldots, (d_m,l_m)@loc \end{array} \right\} \end{aligned}$$

where $u \in U^{DB} \wedge d \in D_{up}$ or $u \in U^{UD} \wedge d \in D_p$, $l, l_1, \ldots, l_m \in L$, $d_1, \ldots, d_m \in D$, and $loc \in Loc$.

- Each action $\phi = (\phi^{in}, \phi^{out}) \in \mathcal{A}_{sdrs}^{(data)}$ is such that

$$\begin{aligned} \phi^{in} &= \{(d_1,l_1)@loc, \ldots, (d_k,l_k)@loc\} \\ \phi^{out} &= \{(d'_1,l'_1)@loc, \ldots, (d'_m,l'_m)@loc\} , \end{aligned}$$

where $loc \in Loc$, $d_1, \ldots, d_k, d'_1, \ldots, d'_m \in D$, and $l_1, \ldots, l_k$, $l'_1, \ldots, l'_m \in L$.

In the above, we assumed for simplicity that the only faulty behaviour (security incident) is data loss, and that a user can only access a single data resource at a time.

Next we consider the dynamic movement of data resources and users.

5.2.2 Data Flow

Data flow security is concerned with the way in which secure information is allowed to flow through a computing system. Intuitively, the flow is considered secure if it adheres to a specified security policy.

Each action $\phi = (\phi^{in}, \phi^{out}) \in \mathcal{A}_{df}$ is such that:

$$\begin{aligned} \phi^{in} &= \{(d_1,l_1)@loc_1, \ldots, (d_k,l_k)@loc_k\} \\ \phi^{out} &= \{(d'_1,l'_1)@loc'_1, \ldots, (d'_m,l'_m)@loc'_m\} , \end{aligned}$$

where $loc_1, \ldots, loc_k, loc'_1, \ldots, loc'_m \in Loc$, $d_1, \ldots, d_k, d'_1, \ldots, d'_m \in D$, and $l_1, \ldots, l_k$, $l'_1, \ldots, l'_m \in L$.

5.2.3 User Flow

The last type of actions concerns the movement of users between locations. Each action $\phi = (\phi^{in}, \phi^{out}) \in \mathcal{A}_{uf}$ is such that:

$$\begin{aligned} \phi^{in} &= \{u@loc\} \\ \phi^{out} &= \{u'@loc'\} \ , \end{aligned}$$

where $u, u' \in U$ and $loc, loc' \in Loc$. Note that it may happen that $u \neq u'$ if the user changes the behaviour status when moving from one location to another.

Note, finally, that in practice the actions of the system can be specified in more convenient way, for example, by using guards and parameters. This is illustrated in the Petri net representations where transitions use guards and arcs are labelled by parameters (variables).

5.2.4 System Security

We have defined general notions related to the syntax and operational semantics of a dynamic data resource system model. It allows one to capture the basic notion of security across the different locations.

Definition 36. *Let DDRS be a dynamic data resource system model as in (5.4). A state s of DDRS is* secure *if $l \leq_{sec} l_{loc}$, for every $(d, l)@loc$ present in s. Moreover, DDRS is* secure *if all its reachable states are secure.* $\Diamond$

That is, a state is secure if all the copies of entities present in the state reside in different locations without causing security violation. One can state a general security policy guaranteeing the security of data flow model. Such a policy is formulated by placing a suitable condition on the actions of the model.

Theorem 3. *Let DDRS be a dynamic data resource system model as in (5.4) such that the following hold:*

- *For every action $\phi \in \mathcal{A}^{(sec)}_{sdrs} \cup A^{(con)}_{sdrs}$, we have:*

$$\coprod\{l_1, \ldots, l_m\} \leq_{sec} l \ .$$

- *For every action* $\phi \in \mathcal{A}_{sdrs}^{(data)}$*, we have:*

$$\coprod\{l'_1, \ldots, l'_m\} \leq_{sec} \prod\{l_1, \ldots, l_k\} \ .$$

- *For every action* $\phi \in \mathcal{A}_{df}$*, we have:*

$$\{loc'_1, \ldots, loc'_m\} \subseteq \{loc_1, \ldots, loc_k\}$$

and, for every $loc \in \{loc'_1, \ldots, loc'_m\}$*, we have:*

$$\coprod\{l'_i \mid loc'_i = loc\} \leq_{sec} \prod\{l_j \mid loc_j = loc\} \ .$$

Then DDRS is secure provided that s_{init} *is secure.* $\Diamond$

The above result can only be applied in specific cases; in general, we need to verify that a given system specification yields a secure system, e.g., by applying a suitable model checking techniques.

5.3 An Example

We will now present an example to illustrate the structure and behaviour of a dynamic data resource system. We assume that two employees, A and B, are on a business trip. Moreover, there are two data resources, $d_1 \in D_p$ and $d_2 \in D_{up}$, which they need to use, and that these data resources are kept on two mobile devices.

During the trip, the employees need to travel between three different locations during the trip, loc_1, loc_2 and loc_3, and access the data whenever they need them. We assume that the behaviour of employee A is captured by $u_1 \in U^{UD}$ in loc_1, and by $u_2 \in U^{DB}$ in loc_2 and loc_3. On the other hand, the behaviour of employee B is captured by u_2 in all three locations.

It is further assumed that there are security levels l_1, l_2, l_3 with the ordering $l_1 \leq l_2 \leq l_3$ and that:

- the security level of the two data resources is l_2; and

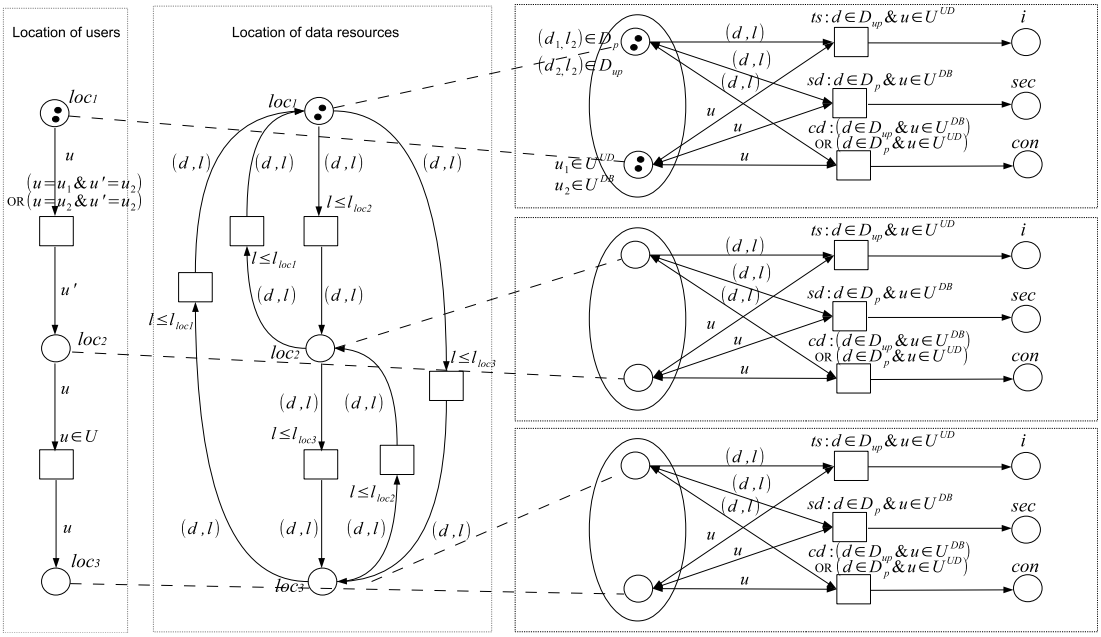


Figure 5.6: A Petri net model of the example system consisting of two users, two data resources, and three locations.

- the security levels of the three locations are as follows: $l_{loc_1} = l_{loc_2} = l_2$ and $l_{loc_3} = l_1$.

The structure and dynamics (possible actions) of the example system are represented by the Petri net in Figure 5.6. The diagram also shows the initial state of the system.

The two employees are represented by two tokens in a place labelled loc_1, one token being u_1 (for employee A) and the other u_2 (for employee B). The leftmost sub-net shows how the users can move between different locations (e.g., it is not possible to move directly from loc_1 to loc_3, but one can achieve this through an intermediate move to loc_2).

The data resources are represented by the two tokens in a different place labelled loc_1. It follows from the security levels of the data resources and locations that data resources can never enter loc_3. Note that the security policy is represented by the guard $l \leq l_{loc_3}$ associated with the transition incoming to the bottom place (labelled loc_3) in the middle sub-net.

Figures 5.7, 5.8 and 5.9 depict three reachable states of the example system:

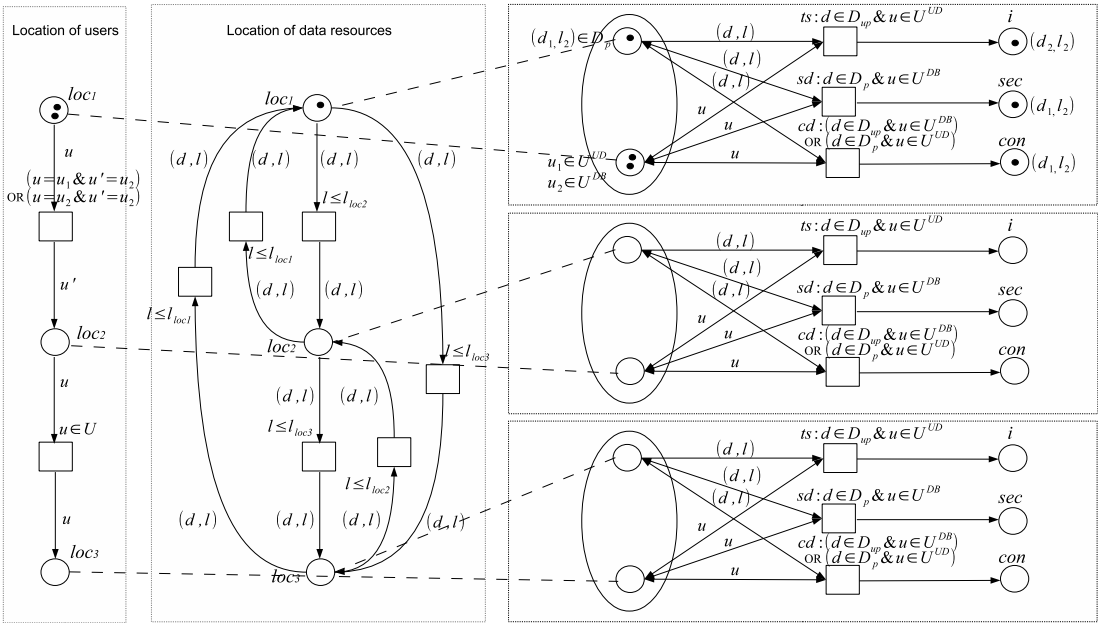


Figure 5.7: A reachable state of the example system with both employees in loc_1.

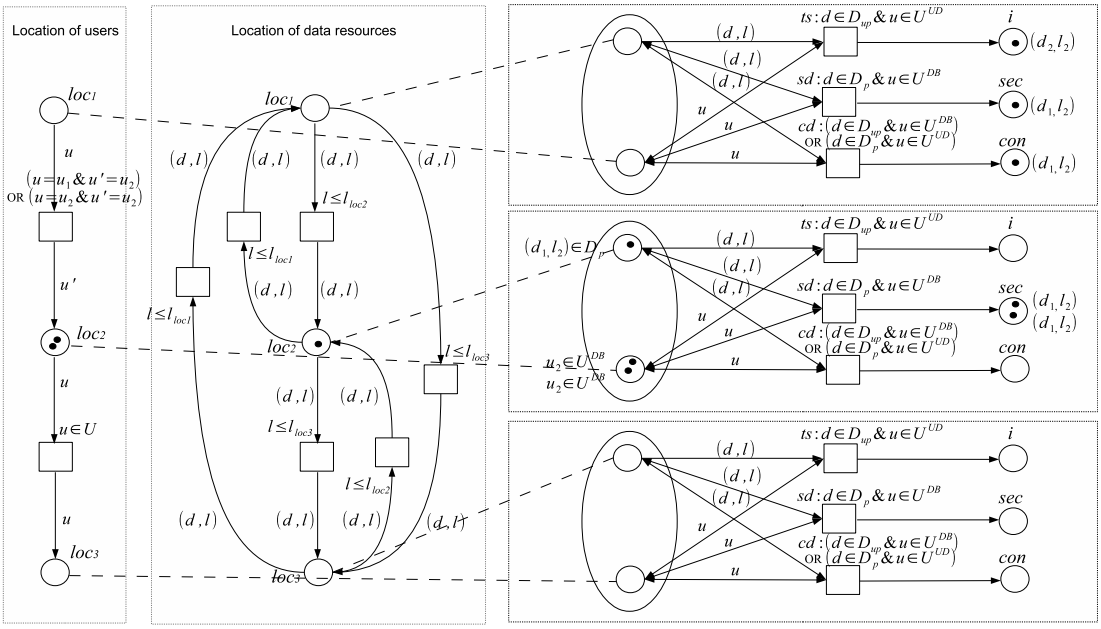


Figure 5.8: A reachable state of the example system with both employees in loc_2.

- Figure 5.7: (i) the undesired behaviour of A at loc_1, combined with the unprotected status of d_2, has led to an incident in which d_2 is lost; (ii) the

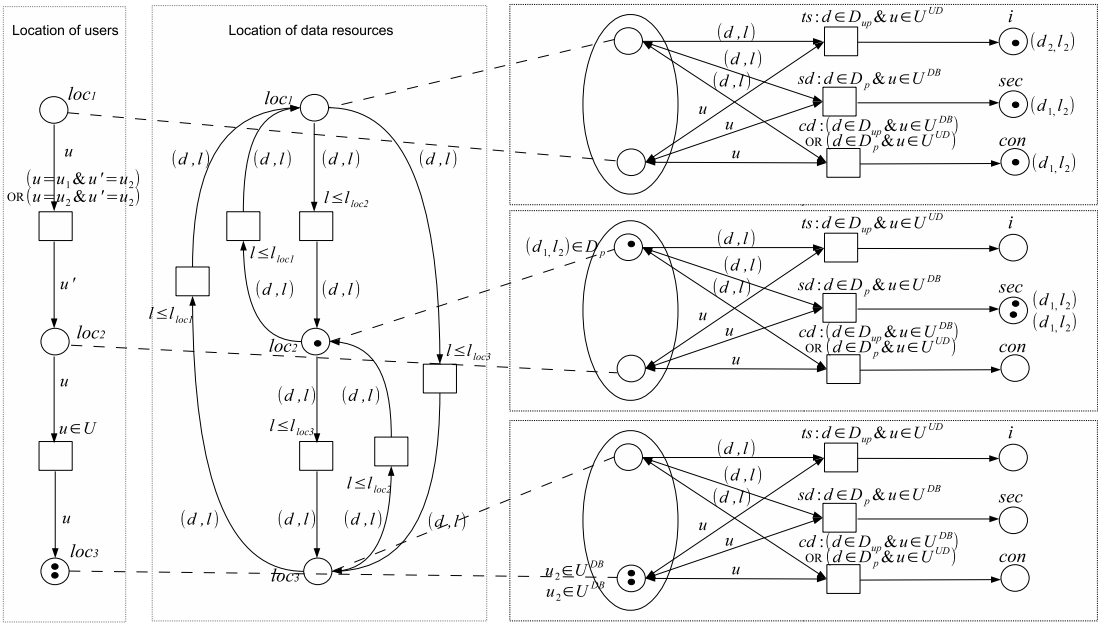


Figure 5.9: A reachable state of the example system with both employees in loc_3.

undesired behaviour of A at loc_1, combined with the protected status of d_1, has led to a controlled interaction; and (iii) the desired behaviour of B at loc_1, combined with the protected status of d_1, has led to a secure interaction.

- Figure 5.8: the desired behaviour of A and B at loc_2, combined with the protected status of d_1, has led to secure interactions.

- Figure 5.9: both A and B can move to loc_3, but d_1 is prevented from doing so by the security policy as discussed above. As a result, the employees cannot interact with d_1 in this location.

5.4 Probabilistic Behaviour

When the users or data resources move from one location to another, their movements might follow probabilistic distribution.

For example, in the system discussed in Section 5.3, users positioned in loc_2 can move to loc_1 or to loc_3. Also, if a data resource with security level l residing

in loc_3 conforms to the security policy, i.e., $l \leq l_{loc_1}$ and $l \leq l_{loc_2}$, the data can move to loc_1 or loc_2.

Assuming that we know the probability distributions for choosing the next locations in such cases, it is possible to turn the Petri net model outlined above into a suitable probabilistic or stochastic Petri net which can then be simulated and evaluated using suitable automated tools.

5.5 Summary

We proposed a system model for data resources in a business organization based on the location of different classes of data resources as well as the users. In this model, the status of the data resources are divided into different classes in order to analyse the existing or planned security policies. We also illustrated how such system model can be represented by a Petri net which can then be used to track and analyse the data resources and users in different locations using suitable tools and methods. The proposed modelling technique would help to move the debate from discussion of claims made by the information security technology vendors to a better (rigorous) understanding of the effectiveness of information security technology and security policies.

Chapter 6

Performance Modelling and Analysis of Information Security Technology

In this chapter, we will consider the trade-off between performance and security when implementing information security technologies in the organization's network. Firstly, an approximate analytical model of the information security system comprising a server and an administrator is proposed and evaluated using queueing theory. A non-productive time (NPT) function for implementing information security technologies is also given. Moreover, a simulation model based on stochastic Petri nets is proposed and evaluated. Secondly, we consider the case of multiple administrators and provide suitable analytical and simulation models which are then compared. Thirdly, a cost function is proposed to analyze the effect of varying the number of administrators in the information system. This part of the study can help an information security manager to estimate the necessary number of administrators providing system support, and the service capacity that has to be guaranteed by the organization in order to satisfy a given number of users.

6.1 Implementing Information Security Technologies

In [57], the authors survey the existing enterprise technologies that control access to confidential digital data (e.g., USB access control solutions, digital rights management software, and disk encryption techniques). The researched technologies use endpoint access control as a means of limiting the maintenance overhead introduced by unauthorized devices. The technologies are installed from a centralized security station, sending client-side installations directly to user workstations. They provide auditing options and prevent outsider access through encryption. The various information security solutions follow a model of centralized control, where access policies are recorded at a single location from which they are passed to end users when they interact with the network, and administrators have the highest access rights. The various information security measures rely on the cooperation of various people and system components, thus carrying them out has an impact on the overall productivity of the organization.

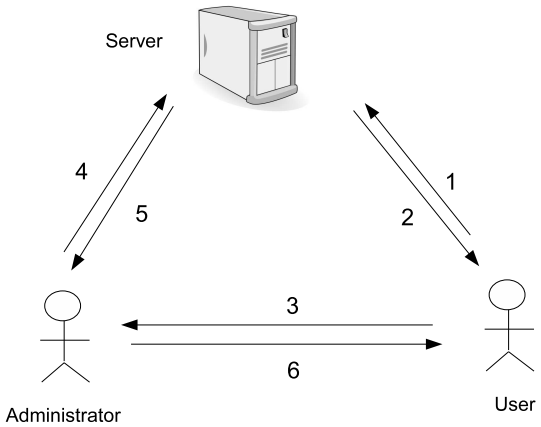


Figure 6.1: Relationships between administrators, users and servers.

Figure 6.1 shows a possible model representing the relationships between users, administrators and servers when implementing information security technologies in the organization's network: (1) When a user tries to access resource, a request is sent to the server. (2) The server attempts to validate the user and, if the user does not pass the authentication procedure, the user is denied the access

the resource. (3) If the access is denied, the user contacts the administrator asking for help. (4) The administrator contacts the server. (5) If the user should be allowed to access the resource, the administrator creates the appropriate access rights for the user, or changes the usage policy for this user in the server. (6) Finally, the administrator sends the access rights to the user.

In our models and experiments presented in the rest of this paper, we have adopted a simplified version of the model depicted in Figure 6.1.

6.2 Queueing Network Model

Figure 6.2 shows an information security system modelled by a simple queueing system with two queue stations (the server and the administrator).

In the system, each user's request is assumed to have a duration specified by a negative exponential distribution with a given mean: $1/r_u$ is the frequency for a user send an access request, $1/r_{ser}$ is the average time it takes the server to serve a user's request, and $1/r_a$ is the average time it takes the administrator to help a non-active user. N is the maximum number of users admitted for processing. If there are more than N requests present, the ones that do not occupy a thread wait in an external FIFO queue.

In the diagram, p $(0 < p \leq 1)$ is the probability that a user can pass the user authentication procedure on the server and become an active user, $1 - p$ is the probability that a user cannot pass the user authentication and becomes an non-active user who needs help from the administrator.

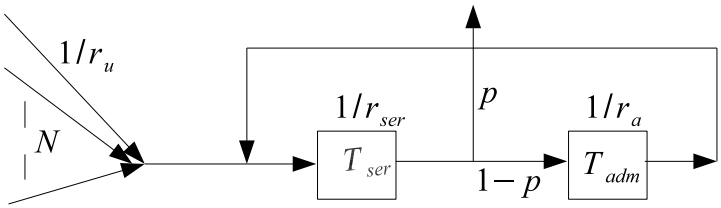


Figure 6.2: A queueing theory model of an information security system.

When the external queue is non-empty, the system behaves like a closed queueing network (Figure 6.3), with N requests circulating between the users and the system.

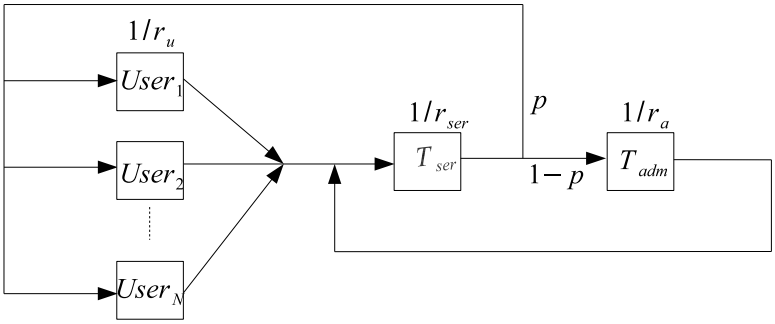


Figure 6.3: A closed queueing theory model of an information security system.

6.3 Approximate Analytical Solution

Let us assume that there are k user requests circulating between the server and the administrator ($k = 1, \ldots, N$). Suppose that the circulation continues for a long time, i.e. the system reaches a steady state with k user requests. Then the server queue would behave like an $M/M/1$ queue with a bounded buffer of size k. The load ρ_{ser} on the server is:

$$\rho_{ser} = \frac{r_u k}{r_{ser} p} \tag{6.1}$$

Using the existing results [14] for the $M/M/1/k$ queue yields the average number of requests in the server:

$$L_{ser} = \frac{\rho_{ser}}{1 - \rho_{ser}} \times \frac{1 - (k+1)\rho_{ser}^k + k\rho_{ser}^{k+1}}{1 - \rho_{ser}^{k+1}} \tag{6.2}$$

The steady state probability Π_k that there are exactly k requests waiting for a response from the server is:

$$\Pi_k = \frac{(1 - \rho_{ser})\rho_{ser}^k}{1 - \rho_{ser}^{k+1}} \tag{6.3}$$

Therefore, the state-dependent throughput of the server when there are k requests in it, T_{ser}, is given by:

$$T_{ser} = (1 - \Pi_k) \times \frac{r_u k}{p} = \frac{1 - \rho_{ser}^k}{1 - \rho_{ser}^{k+1}} \times \frac{r_u k}{p} \tag{6.4}$$

The probability U_{ser} that the server is busy, given that there are k requests in the system is:

$$U_{ser} = \frac{T_{ser}}{r_{ser}} = \frac{1 - \rho_{ser}^{k}}{1 - \rho_{ser}^{k+1}} \times \frac{r_u k}{r_{ser} p} \tag{6.5}$$

The average response time, W_{ser}, of a request that is admitted into the server can be found from Little's theorem:

$$W_{ser} = \frac{L_{ser}}{T_{ser}} \tag{6.6}$$

The entire system is in a steady state. Thus, the load on the administrator, ρ_{adm} is:

$$\rho_{adm} = \frac{(1-p) r_u k}{r_a p} \tag{6.7}$$

Therefore, the state-dependent utility of the administrator when there are k requests in the system, T_{adm}, is given by:

$$U_{adm} = \frac{1 - \rho_{adm}^{k}}{1 - \rho_{adm}^{k+1}} \times \frac{(1-p) r_u k}{r_a p} \tag{6.8}$$

and the average number of requests on the administrator is given by [14]:

$$L_{adm} = \frac{\rho_{adm}}{1 - \rho_{adm}} \times \frac{1 - (A+1)\rho_{adm}^{A} + A\rho_{adm}^{A+1}}{1 - \rho_{adm}^{A+1}} \tag{6.9}$$

where, $A = (1-p)k$.

The average response time, W_{adm}, of a request that is admitted into the administrator can be found from Little's theorem:

$$W_{adm} = \frac{L_{adm}}{T_{adm}} \tag{6.10}$$

The non-productive time (NPT) in the organization is the average number of requests in the server and administrator in an interval of time:

$$NPT = (\frac{L_{ser}}{r_q} + L_{ser} + L_{adm}) \times l \tag{6.11}$$

where l is the period of time users spend in the system, and $1/r_q$ is the average

time taken by a user to send an access request.

6.3.1 Comparing Analytical and Simulation Results

We have compared the above approximate analytical solution with simulation results obtained using the Möbius system [55]. The performance of the system under different loading conditions and parameter settings was examined in a series of numerical and simulation experiments. The main purpose of the simulations was to evaluate the accuracy of the analytical solution and, at the same time, to validate the modelling technique based on stochastic Petri nets.

In this model, we will increase the number of users up to tens of thousands, using a discrete approximation to keep the state space limited. If we assume 100 active users circle around in the model, each representing a group of users as determined by a model multiplier. By incorporating the multiplier correctly in the various transition rates, we can approximate the behaviour of a system with tens of thousands of users by a model that has less than one million states. Arguably, one million states is still a considerable amount, but easily manageable with Möbius [55].

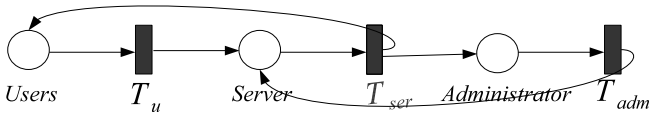


Figure 6.4: A stochastic Petri net model of an information security system. Informally, the middle transition produces a token with in one of the two output places with probabilities p and $1 - p$, respectively.

Figure 6.4 shows the structure of a stochastic Petri nets model representing information security scenario we discussed above. The model consists of three places and three timed transitions. Timed transitions are associated with random exponential distributed firing delays. Authorized *Users* try to access protected resources every $\frac{1}{r_u}$ unit time, each attempt taking $\frac{1}{r_q}$ units of time. We use T_u to control the frequency of access requests sent to the *Server* by a user. The time taken to access the protected resources is given by T_{ser}. If the user can pass the authentication process, then the user can use the resource, but if the user cannot access the resource, the user has to contact the *Administrator* for help. After

obtaining such help (the time taken is given by T_{adm}), the user can try to access the resource again.

The behaviour of the model can be measured by the *impulse rewards model* and *rate rewards model*, which are supported by the Möbius software. The throughput of a transition is computed according to the formula which is described in Section 2.3.2 $\sum_{a \in Tr} \mathcal{C}_a N^a_{[t,t+l]}$. The number of tokens in sets of places is computed according to the formula $\sum_{v \in \mathcal{P}(Pl,\mathbb{N})} \mathcal{R}_{(v)} M^v_{[t,t+l]}$. The time scale of the model is expressed in minutes, i.e. when we run the model one time unit in Möbius represents one minute in real working time.

To measure the throughput of the server, the throughput of a transition per unit of time T_{ser} was computed in average interval of time, and then we could calculate the utility of the server by using the equation (6.5). To measure the throughput of the administrator, the throughput of the transition per unit of time T_{adm} was computed in average interval of time, and then we could calculate the utility of the administrator by using the equation (6.8).

As the number of the parameters is quite high, some of them were kept fixed throughout. These were: the probability that a user passes the authentication procedure ($p = 0.7$); the average time a user needs to send an access request ($\frac{1}{r_u} = 100$); and the average time it takes a user to send a request ($\frac{1}{r_q} = 0.25$).

Figure 6.5 shows the utility of the server against the number of users for both the simulation and approximate analytical approaches for various values of r_{ser} and r_a. Increasing the value of r_{ser} and r_a corresponds to increasing the speed of user access to the resource.

Figure 6.6 shows the utility of the administrator against the numbers of the users for both the simulation and approximate analytical approaches for various values of r_{ser} and r_a. The results show that there are obvious benefits from increasing the server and administrator speed. If the target utilization of the administrator is 0.72, with $r_{ser} = 15$ and $r_a = 4$ one can serve at most 640 users. After increasing the server rate and administrator's service rate to respectively $r_{ser} = 30$ and $r_a = 8$, the utility of the administrator can increase to 0.71, which is close to the target with 1280 users.

Let us now consider one year of work after the deployment of the information security technologies in the network system of an organization, i.e., we consider

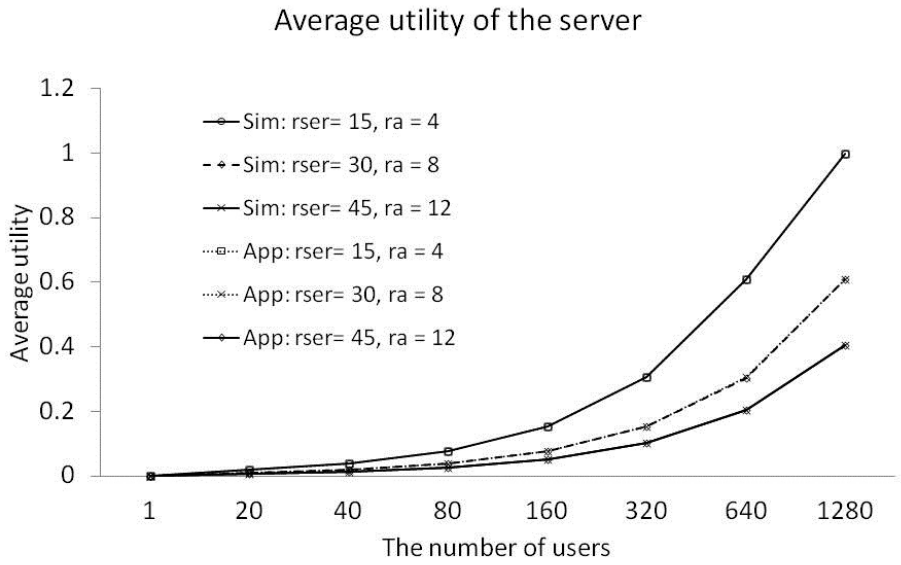


Figure 6.5: Utility of the server w.r.t. the number of users in the system. $p = 0.7, \frac{1}{r_u} = 100$. The utility increases significantly when the number of users served by the server and administrator increases.

96000 time units in the stochastic Petri net model (this corresponds to 40 weeks of work, each working week having 40 working hours). To measure the NPT, the time users spend in any place other than *Users* is computed. The NPT also includes the time the user takes for sending an access request ($\frac{1}{r_q}$).

Figure 6.7 shows the NPT of the system w.r.t. the number of users for various values of r_{ser} and r_a. Increasing the value of r_{ser} and r_a is equivalent to increasing the speed of users access to the resource. The NPT includes: the time spent on sending an access request and authentication procedures, and the time spent on waiting for a response from administrator.

If the target maximum NPT is 120000 time units, one can try to build the system with $r_{ser} = 15$ and $r_a = 4$. Then the system can serve 320 users and the NPT is around 102653 time units which is below the target. However, if we increase the rates to $r_{ser} = 30$ and $r_a = 8$, the system can serve 640 users and the NPT is around 102669 time units.

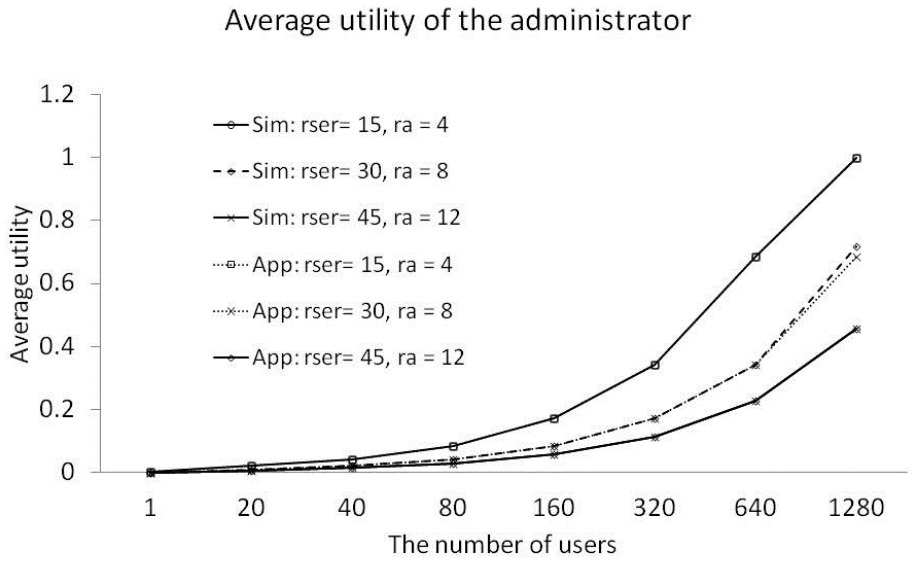


Figure 6.6: Utility of the administrator w.r.t. the number of users in the system. $p = 0.7$, $\frac{1}{r_u} = 100$. The utility increases significantly when the number of users served by the server and administrator increases.

6.4 Multiple Administrators

In the above, we proposed an approximate analytical model for implementing information security technologies. We considered one server and one administrator. In what follows, we consider multiple administrators who provide help with access control problems.

We assume that there are K administrators, each of which can serve one user request at a time, independently of the others (Figure 6.8). We want to know if it is beneficial to increase the number of administrators, or to increase the operational speed of the administrators. It is well known that for an $M/M/K$ queue, it is preferable to have one administrator serving at the rate μ rather than K administrators serving at the rate μ/K [14]. This is because if there are fewer than K requests in the queue, then some of the administrators will be idle, thus reducing the overall service rate.

Consider first the administrator subsystem, with j requests circulating between the server and administrators, $j = 0, 1, \ldots, (1-p)N$, where N is the

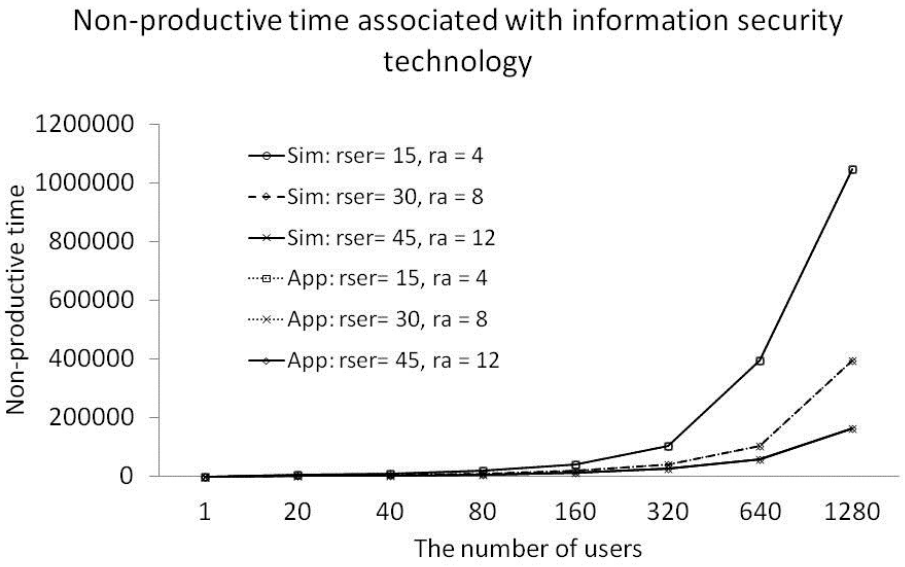


Figure 6.7: Total non-productive time (NPT) associated with the deployment of information security technologies. $p = 0.7$, $\frac{1}{r_u} = 100$. NPT increases significantly when the number of users served by the system increases.

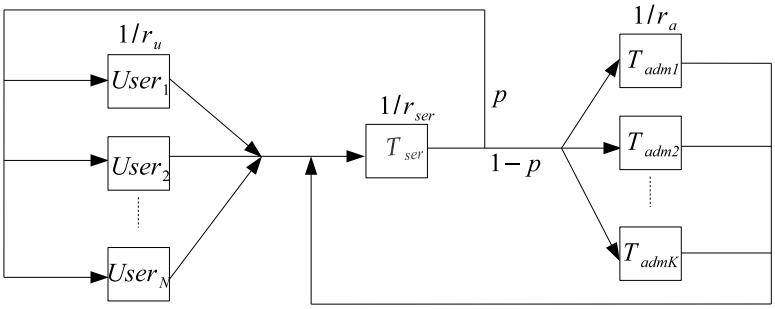


Figure 6.8: A closed queueing theory model of an information security system, with N users and K administrators.

maximum number of user requests, and p is the probability that a user can pass the user authentication on the server.

Suppose that the circulation continues for a long time and the subsystem reaches a steady state with j requests. If there is one administrator in the system and j user requests in the subsystem, the approximation becomes an $M/M/1/./A$

queue ($A = (1-p)N$). Hence the balance equations become [14]:

$$(A-j)r_u\Pi_j = r_a\Pi_{j+1}, \quad 1 \leq j \leq A \tag{6.12}$$

where Π_j is the steady state probability that there are exactly j user requests waiting for a response from the administrator.

Now we increase the number of parallel administrators in the model. The model becomes an $M/M/K/./A$ queue, where K is the number of administrators. Therefore, the balance equations become [14]:

$$(A-j)r_u\Pi_j = (j+1)r_a\Pi_{j+1}, \quad 0 \leq j < K \tag{6.13}$$

$$(A-j)r_u\Pi_j = Kr_a\Pi_{j+1}, \quad K \leq j < A \tag{6.14}$$

We can calculate Π_0:

$$\Pi_0 = \left[\sum_{j=0}^{K-1} \frac{A!\rho^j}{(A-j)!j!} + \sum_{j=K}^{A} \frac{A!\rho^j}{(A-j)!K!K^{j-K}}\right]^{-1} \tag{6.15}$$

The average queue length can then be calculated by [14]:

$$L_{adm} = \sum_{j=1}^{A} j\Pi_j \tag{6.16}$$

$$= A!\Pi_0 \left[\sum_{j=1}^{K-1} \frac{\rho^j j}{(A-j)!j!} + \sum_{j=K}^{A} \frac{\rho^j j}{(A-j)!K!K^{j-K}}\right] \tag{6.17}$$

Each of the users submits requests to administrators at the rate $\frac{(1-p)r_u}{p}$. Therefore, the throughput T_{adm} is [14]:

$$T_{adm} = (A - L_{adm})\frac{r_u A}{p} \tag{6.18}$$

and the average response time of administrators, W_{adm}, becomes:

$$W_{adm} = \frac{A}{T_{adm}} - \frac{p}{r_u A} \tag{6.19}$$

The non-productive time (NPT) in the organization can be calculated using the equation (6.11).

6.4.1 Comparing Analytical and Simulation Results

We again used the Möbius software [55; 60] to simulate the behaviour of the approximate analytical model, and to compare the simulation and analytical results.

Figure 6.9 shows the structure of a stochastic Petri net for the analytical model we have just discussed, which consists of five places, four timed transitions, and one instantaneous transition. Authorized *Users* try to access resources every $\frac{1}{r_u}$ time units during working hours (the time taken is given by T_{ser}). If the user cannot access the resource, administrators are contacted for help. Here we use T_{help} to control the probability with which the users are handled by different administrators. There are eight administrators (*Adm1*, ..., *Adm8*), each of which can serve one user request at a time, independently of the others. The throughput and average response time can be computed as in Section 6.3.1.

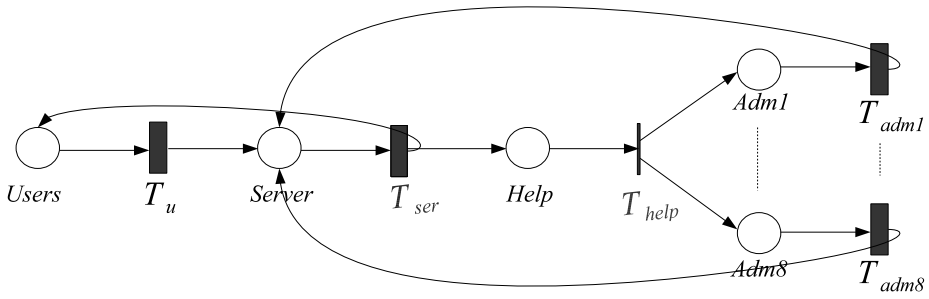


Figure 6.9: A stochastic Petri net model of an information security system with eight administrators. Note that the 'thin' transition with eight output places takes no time to execute, i.e. it is instantaneous.

As before, some parameters were kept fixed: the probability that a user passes the authentication procedure ($p = 0.7$); the frequency with which a user sends

access requests ($\frac{1}{r_u} = 100$) (i.e., a user sends a request every 100 time units); and the average time it takes for a user to send an access request ($\frac{1}{r_q} = 0.25$).

Figure 6.10 shows the average load on the administrators against the number of users for $K = 8$ with $r_a = 0.5$, and $K = 4$ with $r_a = 1$.

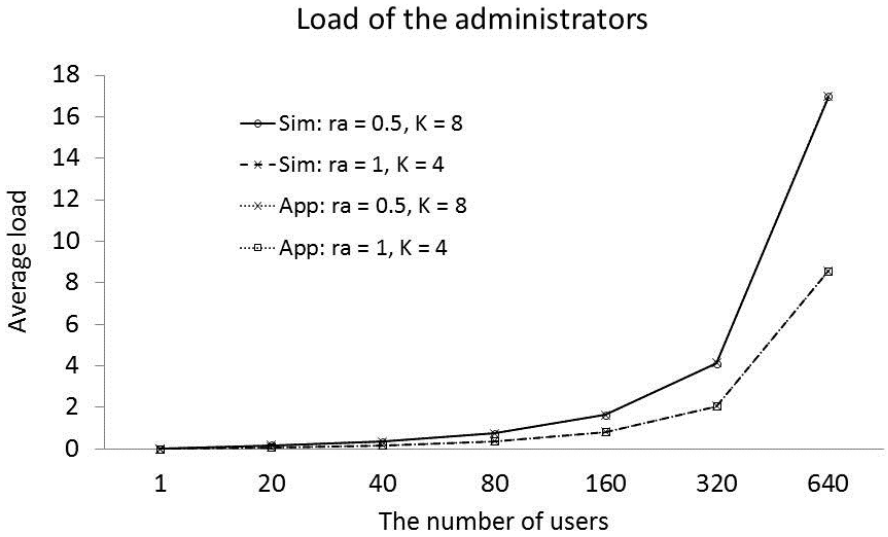


Figure 6.10: Average load on the administrators w.r.t. the number of users in the system. $p = 0.7, \frac{1}{r_u} = 100, r_{ser} = 15$.

Figure 6.11 shows the average response time of the administrators w.r.t. the numbers of user for $K = 8$ with $r_a = 0.5$, and $K = 4$ with $r_a = 1$.

Consider now one year of the deployment of the information security technology in the network system, i.e., 96000 time units in the stochastic Petri net model.

Figure 6.12 shows the NPT of the system w.r.t. the numbers of users for $r_a = 0.5$ with $K = 8$, and $r_a = 1$ with $K = 4$. NPT increased significantly when the service speed of the administrators is slow; in other words, increasing the speed of the administrators reduces the NPT of the staff members in the organization.

Figure 6.13 shows the NPT of the system w.r.t. the number of administrators with 400 users in the system. The system reaches a steady state with five admin-

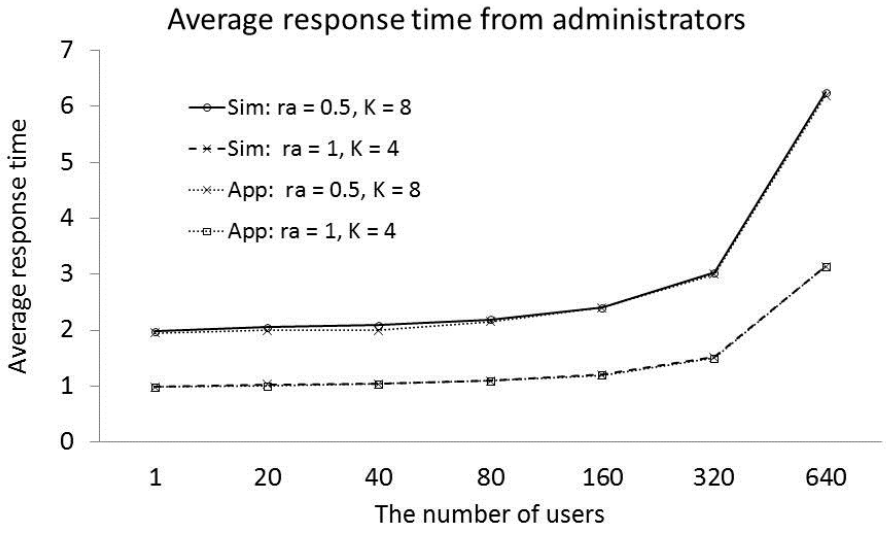


Figure 6.11: Average response time of the administrators w.r.t. the number of users in the system. $p = 0.7, \frac{1}{r_u} = 100, r_{ser} = 15$.

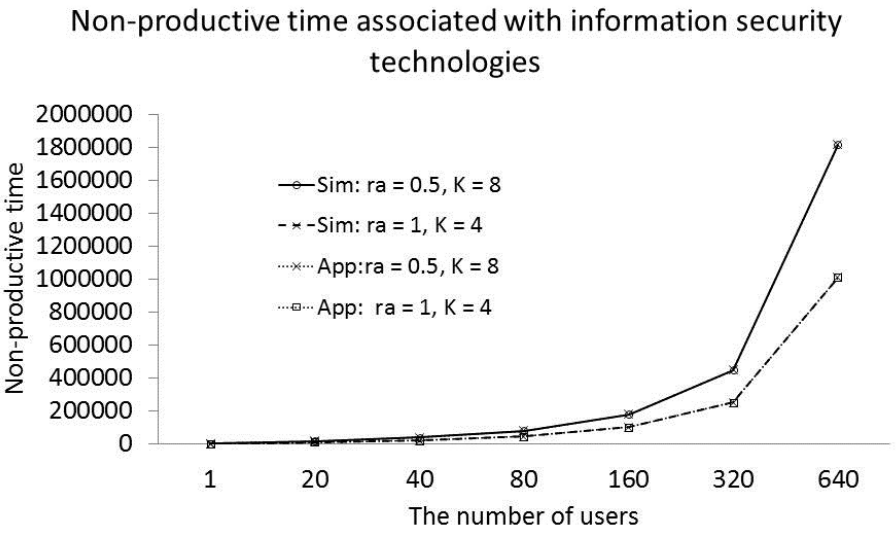


Figure 6.12: Total non-productive time (NPT) associated with the deployment of information security technologies. $p = 0.7, \frac{1}{r_u} = 100, r_{ser} = 15$.

istrators in the system. Increasing the number of administrators will reduce the NPT in the organization. However, the administrators will often be idle in such a case.

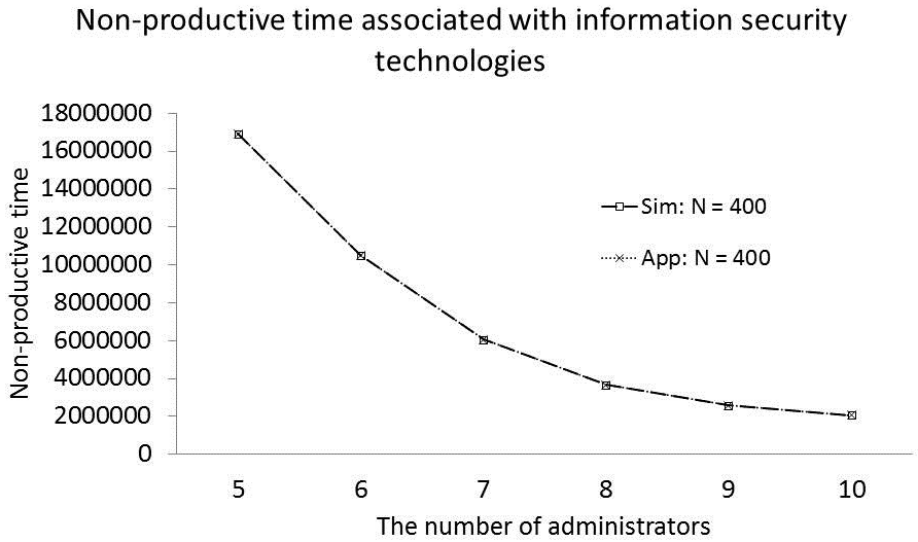


Figure 6.13: Total non-productive time (NPT) associated with the deployment of information security technologies. $p = 0.7, \frac{1}{r_u} = 100, r_{ser} = 15, N = 400, r_a = 0.25$.

6.5 The Cost Model for Administrators

Now we introduce a cost function which needs to be optimized. This function is based on the assumption that there is a cost of the users' waiting time and a competing cost of providing resources, e.g., salaries of administrators, and administrators' training expenditure. This gives rise to the following simple cost function [14; 61]:

$$C = c_1 L_{adm} + c_2 K r_a, \quad c_1, c_2 \geq 0 \tag{6.20}$$

The cost rates $c_i (i = 1, 2)$ are non-negative constant, which are dependent on the particular system, or depend on the type of quality of service contract that is in place. If c_1 is large, in order to keep the total cost C low, L_{adm} should be small

[14]. At the same time, if c_2 is large, in order to keep the total cost C low, Kr_a should be small [14]. However, the coefficients $c_i(i = 1, 2)$ are not necessarily optimal, because the load of the administrators also plays a key role in determining the best strategy, since the service time and the number of administrators also influence the load of the administrators. In general, if the organizations want to improve the responsiveness of the system, they would increase c_1, and if they want to minimize running costs, they would increase c_2.

6.5.1 Analytical Results

We now illustrate the cost function we proposed above using the previous analytical results. Figure 6.14 shows the cost w.r.t. the number of users. It is clear that under the parameter values with $r_a = 0.25$, the cost rises rapidly at around 320 users, which is the approximate maximum capacity the administrators can handle before the performance starts to degrade. Under the parameter values with $r_a = 0.5$, the cost rises at around 640 users. Therefore, doubling the service rate from $r_a = 0.25$ to $r_a = 0.5$ effectively doubles the capacity of the system. In a small system, when $N < 320$, the cost function is dominated by $c_2 K r_a$. Therefore, the cost is greater for faster administrators. The reason is that the administrators will often be idle, and the system is not making efficient use of resources.

Figure 6.15 shows the cost w.r.t. the number of users. It is easy to see, under the parameter values with $r_a = 0.25$, the cost rises rapidly at around 320 users for all cases ($c_2 = 0.1, 1, 10$). In the above pictures, 320 is critical point because of the parameters we put into the model, if the value of the parameters change, 320 would not be the critical point.

Figure 6.16 shows the cost w.r.t. the number of administrators in the system. In this experiment, the number of users is fixed. The larger c_2 results in a decreasing cost before the optimal point and an increasing rate after the point. For 400 users, in the case of $c_2 = 1$ the optimal value is $K = 17$, which gives the minimal cost of 9.62. In the case of $c_2 = 10$ the optimal value is $K = 10$, which gives the minimal cost of 35.29. When information security managers make the trade-off between security and cost, the balance point here is the value of K

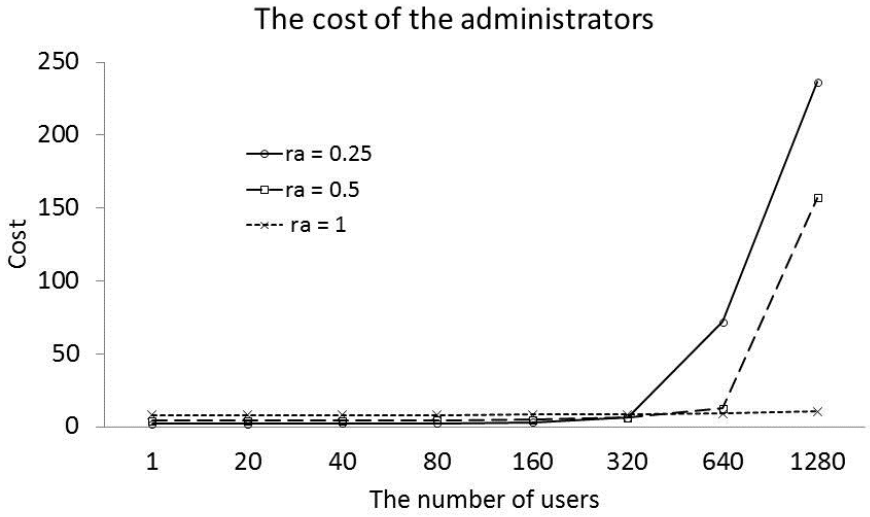


Figure 6.14: The cost w.r.t. the number of users calculated by the queueing network model. $p = 0.7, \frac{1}{r_u} = 100, r_{ser} = 15, K = 8, c_1 = 0.5, c_2 = 1$.

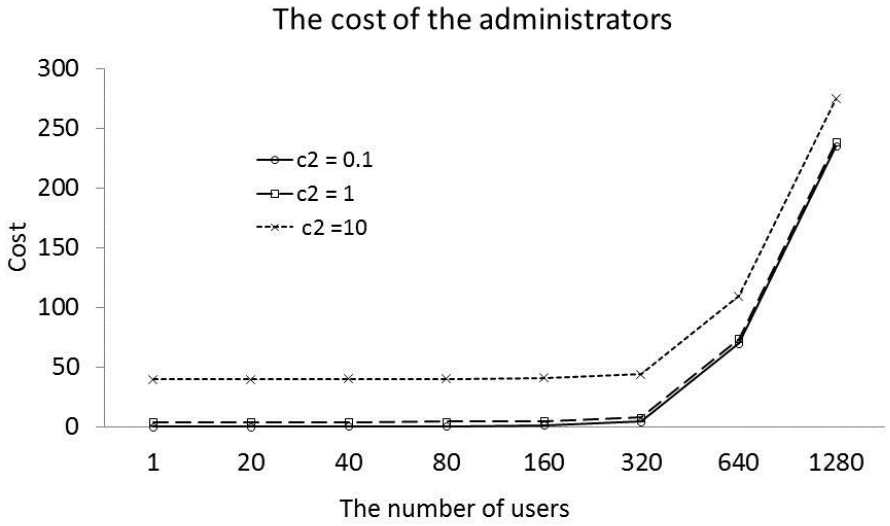


Figure 6.15: The cost w.r.t. the number of users calculated by the queueing network model. $p = 0.7, \frac{1}{r_u} = 100, r_{ser} = 15, K = 8, r_a = 0.25, c_1 = 0.5$.

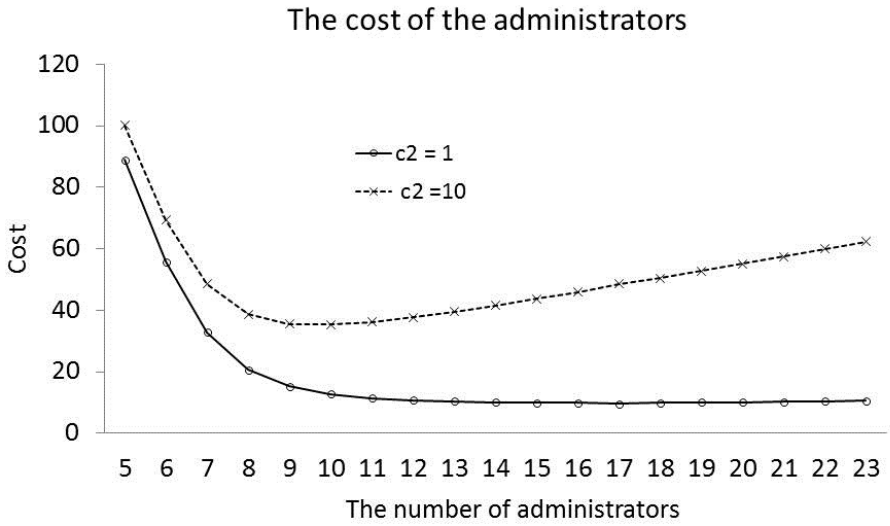


Figure 6.16: The cost w.r.t. the number of administrators. $p = 0.7, \frac{1}{r_u} = 100, r_{ser} = 15, N = 400, r_a = 0.25, c_1 = 0.5$.

where the cost is minimal. Therefore, the security manager could choose $K = 17$ and $K = 10$ when $c_2 = 1$ and $c_2 = 10$, respectively.

6.6 Sensitivity Analysis of Productivity Loss in the Organization

As above we can see non-productive time (NPT) associated with enterprise information security technologies is related to a variety of parameters, which are affected by the security policies and user behaviour in the organization. Since these parameters have an impact on the organization efficiency loss, it is of critical business importance for corporations to find out which parameter result in a large contribution to the overall output variability, so that enterprise-wide security budget can be optimized to these factors in order to minimize the cost [62]. Sensitivity analysis is particularly useful in this situation.

The methodology of sensitivity analysis is well developed and widely used in project development and business strategy [63; 64]. In this part, we will analyze

the relative importance of each parameters that have an impact on the organization efficiency by applying sensitivity analysis to the stochastic Petri nets (or queueing theory) model.

6.6.1 The Approach

To get the aim of the sensitivity analysis, three main steps are proposed:

Step 1: A base case model should be built to reflect the business process by implementing an information security technology. The base case should incorporate the best bet values of input parameters, following an initial run with a base case model, a belief about the optimal strategy can be formed [66]. This belief is based on the modeller's perceptions of probability distribution of profit for the preferred strategy.

Step 2: Key parameters of the base case model should be identified. The domain for each of the input parameters of the model should include base case value, maximum value, and minimum value.

The input key parameters of the base case model is a finite set: $X = \{x_1 \dots x_n\}$. We use x_i^{bas} denotes the base case value of input parameter x_i, x_i^{max} denotes the maximum value of x_i, and x_i^{min} denotes the minimum value of x_i. In addition, the output of the base case is denoted by y^{bas}.

Step 3: When analysing the sensitivity of x_i, the parameter is varied by using its maximum or minimum value, while leaving all other parameters at base values. We calculate the output of the model, and assess the influences of each input parameter relative to the output of the model[1]. The sensitivity of x_i can be calculated as:

$$Positive\ part: \quad +S_{x_i} = \frac{y_i^{max} - y^{bas}}{x_i^{max} - x_i^{bas}}$$

$$Negative\ part: \quad -S_{x_i} = \frac{y^{bas} - y_i^{min}}{x_i^{bas} - x_i^{min}}$$

[1]The units of measurement of different parameters might not be comparable, so there cannot be absolute scopes with respect to changes in different parameters. This problem can be overcome by measuring the percentage change in the output, divided by the percentage change in the input parameter. Therefore, the sensitivity coefficient is the ratio of the change in output to the change in input while all other parameters remain constant

where, x_i^{max} and x_i^{min} correspond to the output values y_i^{max} and y_i^{min}, respectively.

6.6.2 Case Study

In this part, enterprise digital rights management (EDRM) is used as a case study (more information can be found in Section 4.2).

6.6.2.1 Define Base Case Model

The model we will use is in Section 6.3: When authorized users try to open documents, the user needs a username and password in order to login to the system, which takes time. If the user can open the document they become an active user. However, if the user cannot pass the document authentication or the user cannot pass the user authentication, the user would contact the administrators for help. After gets help from the administrators, the user would try to use the document again.

The input parameters for the base case model are in Table 6.1. The values of these parameters are gathered using interviews, experiments, and literature reviews; more details about these data can be found in Section 6.6.2.2. The system reaches steady state under the parameters in the table, based on equation (6.1) and (6.7):

$$\begin{aligned} \rho_{ser} &= \frac{r_u k}{r_{ser} p} = 0.00007 < 1 \\ \rho_{adm} &= \frac{(1-p) r_u k}{r_a p} = 0.0132 < 1 \end{aligned}$$

where, $\frac{1}{r_u} = 120$, $k = 50$, $r_{ser} = 6000$, $r_a = 0.222$, $p = 0.993$.

We measure the NPT, using equation (6.11), within one year of the deployment of EDRM in the organization's network system, i.e., 96000 time units in the stochastic Petri nets model. A middle-size corporation that has 50 authorized users will incur about 15.271 hours of NPT, which is associated with the total waiting time lost on the information help desk, and the total authentication time loss of the corporation.

Table 6.1: The input parameters of the base case model, which can be gathered using interviews, experiments and literature reviews.

Parameters	Value
Number of authorized users in the corporation	50
Number of documents, a user might use every day on average	4
Number of normal working hours per day	8
Number of working weeks per year	40
Average time service need to serve each user	0.01 seconds
Average time administrators need to help each user	4.5 minutes
Average time users need to spend to pass user authentication	7.33 seconds
Percentage of time when authorized users experience a login system failure and attempt ask administrator for help	0.70 %
Number of administrators in the organization	1

6.6.2.2 Define Key Parameters

We now need to understand how the parameters affect the NPT of the corporation.

There are nine input parameters in the model (Table 6.1). The number of authorized users in the corporation, the number of documents that an authorized user might use every day on average, the number of normal working hours per day, and the number of working weeks per year are invariant input parameters of the model. These four parameters depend entirely on the size and business type of different corporations. Therefore, the values of these parameters are determined by literature based assumptions.

With one administrator in the system, the system can reach steady state; therefore, the number of administrator also set as invariant parameter.

The other four input parameters of the model are variable numbers. They can be changed by the security policies of the organization. These parameters could be adjusted independently. Table 6.2 lists the four parameters that will be

Table 6.2: The value of each input parameter in the model for the sensitivity analysis. The base value incorporates the best bet values of the parameters, and the parameters are varied by their maximum and minimum values.

Parameters	Highest value	Base value	Lowest value
1. Average time service response time	0.14 seconds	0.01 seconds	0.001 seconds
2. Average time administrators need to help each user	10 minutes	4.5 minutes	1 minutes
3. Average time users need to spend to pass user authentication	17 seconds	7.33 seconds	4 seconds
4. Percentage of time when authorized users experience a login system failure and attempt ask administrator for help	26.1%	0.7 %	0%

investigated in the sensitivity analysis.

In Table 6.2, parameters 1 & 4 are based on the literature review; parameter 2 is based on an interview; parameters 3 is based on an experiment.

Parameter 1: The values of this parameter are the measurements from a real Google service [67]. The Google search system updates query results interactively as the user types, performing the search and showing the results within a few tens of milliseconds. A request fan out from a root to a large number of leaf servers and merging responses via a request-distribution tree. From a real Google service [67], each server typically responds in 10 ms ($x_1^{bas} = 0.01sc$). The maximum response time is 140 ms ($x_1^{max} = 0.14sc$), and the minimum response time is 1ms ($x_1^{min} = 0.001sc$).

Parameter 2: This interview is from Newcastle University. All the students in the university have their own username and password to access facilities, computers, and network in the campus. When students experienced username and password issues, they would go to the University's information help desk ask for help.

In the interview, the technical staff member in the information help desk claimed that: It takes 1 or 2 minutes for the staff change a student's password, after changing the password, the password is available for use immediately. Creating a new account for a student would take 5 or 10 minutes, which would also be available for immediate use. If the username and password of enterprise Digital Rights Management product (EDRM) belong to the campus domain, the procedure to creating or changing the username and password would be the same as normal password changing and creating procedure.

In our study, the following assumption is made: after help desk staff change or create a username and password for a user, the user can use the username and password to open protected documents immediately; the values of the parameters are uniform in distribution, and then the base case of *parameter 2* is: $x_2^{bas} = \frac{1+2+5+10}{4} = 4.5$, which means the average time technical staff spend to change or create a username and password is 4.5 minutes. The maximum time cost is 10 minutes ($x_2^{max} = 10$); and the minimum time cost is 1 minutes ($x_2^{min} = 1$).

Parameter 3: The participants in the experiment were 30 students in Newcastle University: 5 PhD students, 3 MSc students and 12 undergraduate students. System logs of their user web-based e-mail accounts were examined in the summer term at a computer cluster and students' offices. Participants used their username and password to log into their web-based e-mail accounts to check their e-mail.

From the username and password setting rules in the university: "The login name is made up of 'a' or 'n' followed by the middle seven characters of the Student Number", "The way to set a password: you need something which is memorable, but not guessable. Think of a phrase of eight words (or more) which contains at least one upper case (capital) letter and at least one number, and then use the initial letters of the phrase" [68].

Out of the 30 participants, the minimum time cost to login into their student account is 4 seconds ($x_3^{min} = 4$). The maximum time cost is 17 seconds ($x_3^{max} = 17$), and the average time cost is 7.33 seconds ($x_3^{bas} = 7.33$). The time cost of each student depended on the length of the username and password, the strength of the password, the character restrictions of the password and the frequency of using and changing the username and password [69; 70; 71].

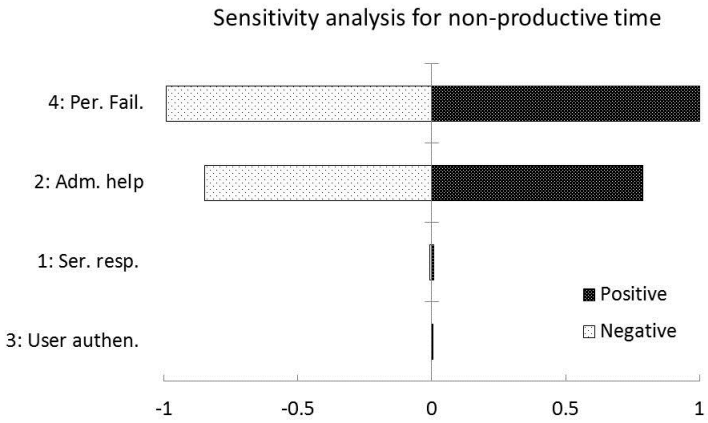


Figure 6.17: The tornado chart is the sensitivity of four key variant parameters in the model. The x-axis is the percentage change from the base case model of the sensitivity analysis. This analysis is associated with the input parameters in Table 6.2.

Parameter 4: This parameter is based on the experiment in University College London. They found that there are approximately 0.7% of login attempts that can be expected to result in a help desk call, where passwords are re-set via the information help desk [72]. The highest percentage of password reminder is 26.1%, and the lowest percentage of password reminder is 0%, i.e., $x_4^{bas} = 0.7\%$, $x_4^{max} = 26.1\%$, and $x_4^{min} = 0\%$.

6.6.2.3 Sensitivity Analysis

Sensitivity is calculated as the ratio between the relative change of model output and the relative change of an input parameter (Section 6.6.1).

Figure 6.17 shows the results of sensitive analysis for three parameters in Table 6.2. It has been found that a small change of *parameter 4* (the percentage of time when authorized users experience a login system failure) contributes most to output variability. *Parameter 3* (the average time users need to login to the system every time when they use the document) gives the smallest relative change in the NPT.

The information security policies determine the value of the paramters and affect the business process of the organization. For example, document classifi-

cation polices associated with the confidentiality will affect the availability of the documents. A strict document classification policy without an effective management of the documents has the potential of delaying the business process when authorized users cannot open a needed document and have to wait for help from administrators. Organizations enforces security policies that ensure maximum protection of their information assets, e.g., frequently change user passwords, and the rules to set the passwords. However, users are disturbed when system policies do not accept their preferred password or oblige them to change it frequently, as users tend to forget their passwords [73; 74]. When users forget their password, they would try different passwords based on their memory or the output of the authentication system. Therefore, good password security policies can help to decrease the NPT by reducing the average time users need to spend in order to pass user authentication.

6.7 Summary

In this chapter, we provided an approximate analytical model for investigating different implementations of information security technologies. We have also provided a corresponding simulation model based on stochastic Petri nets. The two approaches have been compared through a series of experiments which demonstrated that the results they can supply are very similar. Hence one can conclude that the approximate analytical solution is sound. Moreover, we can conclude that the simulation technique based on stochastic Petri nets can be relied upon when it comes to the evaluation of, e.g., productivity loss caused by the introduction of security technologies.

We proposed functions to estimate the non-productive time (NPT) in an organization resulting from the implementation of security technologies, and the cost function for the administrators in the information help desk.

Queueing theory was used to analyse the implementation of information security technologies, and stochastic Petri nets were used to simulate the approach. The effect of several controllable parameters on the performance of the system was examined in a series of numerical and simulation experiments.

Sensitivity analysis is used to determine which of the parameters exerts the

most influence on the NPT of an corporation, in order to help information security managers to balance the weight of information security expenses, and make well informed security investment decisions. Proper data collection methodologies were used, based on the nature of each input parameter, and EDRM was used as a case study. In this study, each parameter is assigned by value. Different values of the input parameters would lead to different results.

Möbius system [55] is used to provide the simulation results. We found that Möbius can not handle more than one million states. Therefore, in these models, we assume 100 active users circle around in the model, each representing a group of users as determined by a model multiplier. By incorporating the multiplier correctly in the various transition rates, we can approximate the behaviour of a system with tens of thousands of users by a model that has less than one million states. However, one million states is still a considerable amount, but easily manageable with Möbius [55].

Chapter 7

Related Work

In this chapter, we discuss related work on quantitative analysis of information flow and information security technologies.

7.1 Information Flow Security

There has been significant activity recently around the questions of how to build a secure service-based system, and how to measure the secure information flow in the programming language. This section gives a overview of the related work and provides a context for our study.

The Bell-LaPadula model [20] is a state machine model which is used to enforce access control in government and military applications. In this model, the entities in the information system are divided into subjects and objects. The notion of a secret state is defined, and it is proven that each state transition preserves security by moving from secret state to secure state by proving the system satisfies the security objectives of the model. A system state is defined to be secure, if the only permitted access modes of subjects to objects are in accordance with a security policy. The paper [75] applied Bell-LaPadula model to work-flow security, and Petri nets were used to model the work-flow. However, [75] does not extend to considering the deployment of resources blocks within a work-flow across a set of computational resources. In [76], the author proposed an extended Petri net formalism called information flow security net (IFS), to

provide a way of modelling information flow security policies expressed through the Petri net structure. Later, in [77], the author proposed another security model, called security Coloured Petri net (SCPN), derived from the IFS net to give a better and more compact representation to efficiently analyse information flow. Then, the paper [18] proposed to partition work-flows over a set of available clouds in such a way that security requirements are met. This approach was based on a multi-level security model that extends Bell-LaPadula to encompass cloud computing. The authors investigated work-flow transformations that are needed when data is communicated between clouds. However, [18] does not consider the concurrency of the events or the execution of tasks in the system.

There exist different methods for the flow-sensitive analysis of programs: In [78], the authors first showed that the data manipulated by a program can be constrained with security level, which naturally assumes the structure of a partially ordered set. Moreover, this partially ordered set is a lattice under certain conditions [5]. The paper [6] presented a lattice-based framework for describing information and flow. In [79], the author first built a formal correspondence between non-interference and mutual information, and established a connection between Shannon's information theory and state-machine models of information flow in computer system. In [80], the authors devised a new information theoretic definition of information flow and channel capacity. In [81], the authors present a notion of soundness for the system that can be viewed as a form of non-interference. Clark, et all. [82; 83; 84; 85] proposed that Shannon's information theory be used to measure the information leakage in imperative programming languages.

Information hiding systems are used to formally analyse the information-hiding properties of protocols and programs. In [16; 24; 25], the authors proposed to use Petri Net modelling technique to specify the opacity of information flow, and adapted opacity to transition system. In [86], the authors presented a probabilistic version of anonymity by using information theory, which is computed using regular expression. In this paper, Crowds Protocol is used as an example. Actions are separated into secret and observable in the general interactive probabilistic information hiding systems. Secret and observable actions can interleave and influence each other. In [87], the authors also studied the problem

of information hiding in systems characterized by the presence of randomization and concurrency.

7.2 Enterprise Information Security Technology

Petri nets have been used for the verification of security requirement [88], for specification of workflows [10], and for security analysis of security policies including Discretionary Access Control policies [89], Mandatory Access Control policies [75; 76; 77], and Role Based Acces Control policies [90; 91]. In [92], the authors used coloured timed Petri nets to implement a workflow authorization model. They used Petri nets to ensure synchronization between authorization flow and the workflow so that subject gain access to the required objects only during the execution of the specific task. Each task is specified with a time interval during which the task must be executed. In [89], the author used Petri nets to analyse dynamic access control in workflow systems. The author proposed assigning a local matrix to each transition to grant rights to subjects (that will execute the transition) only to data being consumed or produced by the transition. When a transition is enabled, subjects have only the access rights which are needed for the execution of this transition. In [75], the author showed how to use Petri nets in order to analyse the information flow in a workflow system where authorizations are granted according to the Bell-LaPadula' model. During the analysis, the objects are assigned all possible security levels in order to enumerate all model states. In [93], the authors proposed a different way to make a system security analysis under Bell-LaPadula's model, by using CPN. The analysis is based on the reachability graph exploration in order to verify some security attributes. In [59], the author proposed a formal model to analyse the system risk. This paper formalized the threats and their impact in the organization. However, [59] does not consider the location of users and their behaviour.

Information security researcher has proposed some methods for formally addressing the problem of data resource security. In [94], the SCRP process calculus is proposed to analyse data resources in different locations. The environment in which the system resides has a stochastic representation using a variety of probability distributions, and a tool is provided to allow one to animate the model as

a discrete event simulation. Such a tool can help an organization to track data resources across different locations. The resulting methodology has been applied to a number of security scenarios, including vulnerability management [95], USB stick usage [96], and identity management [97]. However, the existing research did not involve user behaviours, and the status of data resources in different locations. Moreover, the tool represents an environment in a stochastic way, using a wide range of probability distributions and queue-like data structures. This can make it very difficult to analyse the risk exposure associated with the choice of information security technologies and policies, as well as the ways in which the threat environment might exploit different levels of risk exposure.

Information security researchers proposed various methods for addressing security investment problems. In [95; 98], the authors used mathematical models and stochastic simulations to examine the effectiveness of security operation processes and protection mechanisms. In [96], the authors proposed to use economic models based on trade-off between information confidentiality, integrity and availability to assess the effectiveness and value of security investment of a system. However, none of them considered the cost on the administrators in the information help desk and the non-productive time (NPT) in the organization. In [57], the authors proposed stochastic Petri nets that can be used to analyse the productivity loss by implementing information security technologies. However, the measurement of the productivity loss is not clear in this study.

Chapter 8

Conclusions and Future Works

8.1 Conclusions

In this Thesis, we analysed the security properties in distributed system. We have studied the information flow security in the cloud computing system, and the data and productivity loss in organizations. The work contained in this Thesis can be summarized as follows.

We have proposed a flow-sensitive security model for cloud computing systems and a security lattice was used to analyse the entities in the system. The opacity and observation of the model were analysed, and probabilistic models proposed to quantify a system's observable behaviour under different security preserving mechanisms. Based on the probabilistic model of opacity, we proposed five alternative definitions of probabilistic opacity for computing systems, each definition capturing a different aspect of quantifying the opacity of a security predicate.

We have proposed QEP to quantitatively evaluate the benefits and costs of implementing information security technologies. EDRM were used as a case study to illustrate our approach. Staff productivity reduction is quantified by non-productive time and translated into monetary terms. Security metrics were defined to measure the impact of EDRM products.

We have formally defined a system model for data resources in the organization based on the location of different classes of data resources as well as users. The status of the data resources are divided into different classes to analyse based on the security policy in the organization. Transition systems are built to track the

data resources and users in different locations. The modelling help us move form the debate from discussion of claims made by the information security technology vendors to a better understanding of the effectiveness of information security technology.

We have analysed the trade-off between security and performance of the administrators when implementing information security technologies. An efficient solution of the information security system comprising the server and administrators was analysed using queueing theory. Non-productive time and cost functions for implementing the information security technologies were also proposed. Stochastic Petri nets models were built to simulate our approach and to evaluate the accuracy of the approximate solution.

Generally, this Thesis provided theory and modelling guideline to help organization make information security related decision.

8.2 Future Works

In this Thesis, we proposed to quantitatively analyse the opacity in the computing system, in this study, we concentrated on the probabilistic behaviour of the system. However, real world system allow input and output, and the behave as an interactive system. Therefore, in the future we would consider to quantitatively analyse of opacity in the interactive systems.

The data resources system we proposed concentrated on dynamic behaviour of data resources and users in different locations, in the future, based on the model we proposed, we would analyse how the threat environment can exploit different levels of risk exposure, in addition, we would analyse the threat associated with the implementing of information security technologies, especially looking at how much data get left in exposure places.

References

[1] A. Adams and M. A. Sasse, "Users are not the enemy," *Commun. ACM*, vol. 42, no. 12, pp. 40–46, Dec. 1999. 1, 54, 55

[2] B. Schneier, *Secrets and lies: digital security in a networked world*. New York: Wiley Computer Publishing, 2000. 1, 54, 55, 56

[3] A. Renyi, *Probability Theory*. North-Holland Publishing Company, 1970. 7

[4] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 1948. 8

[5] D. E. Denning, "A lattice model of secure information flow," *Commun. ACM*, vol. 19, pp. 236–243, May 1976. 10, 11, 80, 119

[6] J. Landauer and T. Redmond, "A lattice of information," in *Computer Security Foundations Workshop VI, 1993. Proceedings*, Jun 1993, pp. 65–70. 11, 26, 80, 119

[7] M. A. Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis, *Modelling with generalized stochastic Petri Nets*. Wiley Series on Parallel Computing, 1995. 11

[8] N. R. Adam, V. Atluri, and W.-K. Huang, "Modeling and analysis of workflows using petri nets," *J. Intell. Inf. Syst.*, vol. 10, no. 2, pp. 131–158, Mar. 1998. 11

[9] K. Salimifard and M. Wright, "Petri net-based modelling of workflow systems: An overview," *European Journal of Operational Research*, vol. 134, no. 3, pp. 664 – 676, 2001. 11

[10] W. M. P. van der Aalst, "The application of petri nets to workflow management," *The Journal of Circuits, Systems and Computers*, vol. 8, pp. 21–66, 1998. 11, 120

[11] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989. 11

[12] K. Jensen, *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*. Springer, 1997. 13, 30

[13] W. H. Sanders and J. F. Meyer, *A unified approach for specifying measures of performance, dependability, and performability*. Springer-Verlag, 1991, vol. 4, pp. 215–237. 17

[14] I. Mitrani, *Probabilistic Modelling*. Cambridge university press, 1998. 17, 18, 19, 20, 21, 95, 96, 100, 102, 106, 107

[15] R. D. Bailliage and L. Mazar, "Using unification for opacity properties," in *Proceedings of the Workshop on Issues in the Theory of Security (WITS04*, 2004, pp. 165–176. 22

[16] J. Bryans, M. Koutny, L. Mazar, and P. Ryan, "Opacity generalised to transition systems," *International Journal of Information Security*, vol. 7, pp. 421–435, 2008. 22, 33, 34, 119

[17] M. A. Law, "Using net present value as a decision-making tool," *Air Medical Journal*, vol. 23, no. 6, pp. 28–33, 2004. 24

[18] P. Watson, "A multi-level security model for partitioning workflows over federated clouds," *Journal of Cloud Computing*, vol. 1, no. 1, pp. 1–15, 2012. 25, 30, 31, 119

[19] J. A. Goguen and J. Meseguer, "Security policies and security models," in *IEEE Symposium on Security and Privacy*, 1982, pp. 11–20. 26

[20] D. E. Bell and L. J. LaPadula, "Secure computer systems: Mathematical foundations," MITRE Corporation, Tech. Rep., Mar. 1973. 26, 118

[21] Biba, "Integrity Considerations for Secure Computer Systems," *MITRE Co., technical report ESD-TR 76-372*, 1977. 26

[22] W. M. P. v. d. Aalst, "Verification of workflow nets," in *Proceedings of the 18th International Conference on Application and Theory of Petri Nets*, ser. ICATPN '97, 1997, pp. 407–426. 30

[23] J. Cortadella, M. Kishinevsky, L. Lavagno, and A. Yakovlev, "Deriving petri nets from finite transition systems," *IEEE TRANSACTIONS ON COMPUTERS*, vol. 47, no. 8, pp. 859–882, 1998. 30

[24] J. W. Bryans, M. Koutny, and P. Y. A. Ryan, "Modelling opacity using petri nets," *Electron. Notes Theor. Comput. Sci.*, vol. 121, pp. 101–115, February 2005. 33, 119

[25] J. Bryans, M. Koutny, L. Mazar, and P. Ryan, "Opacity generalised to transition systems," in *Formal Aspects in Security and Trust*, ser. Lecture Notes in Computer Science, 2006, vol. 3866, pp. 81–95. 33, 119

[26] T. M. Cover and J. A. Thomas, *Elements of information theory*. New York, NY, USA: Wiley-Interscience, 1991. 49

[27] A. Dolya, "Internal it threats in europe 2006," vol. 2010, no. November 1. the InfoWatch Analytical Center, 2006. 53

[28] R. A. Huebner and M. M. Britt, "Analyzing enterprise security using social networks and structuration theory," *Journal of Applied Management and Entrepreneurship*, vol. 11, no. 3, p. 6878, 2006. 54

[29] E. Humphreys, *Information security risk management*. British Standards Institution, 2010. 54, 56, 61

[30] R. Layard and S. Glaister, *Cost-benefit analysis*, 2nd ed. Cambridge: Cambridge university press, 1994. 54, 58

[31] T. H. Davenport and J. E. Short, "The new industrial engineering : information technology and business process redesign," *Sloan Management Review*, vol. 31, pp. 11–27, 1990. 55, 57

[32] A. Foroughi, M. Albin, and S. Gillard, "Digital rights management: a delicate balance between protection and accessibility," *Journal of Information Science*, vol. 28, no. 5, pp. 389–395, 2002. 55

[33] H. Cavusoglu, B. Mishra, and S. Raghunathan, "The effect of internet security breach announcements on market value: capital market reactions for breached firms and internet security developers," *Journal of Electronic Commerce*, vol. 9, no. 1, 2004. 56

[34] A. Gary, J. Curtis, and H. Halper, "Quantifying the financial impact of it security breaches," *Information Management & Computer Security*, vol. 11, no. 2, pp. 74–83, 2003. 56

[35] K. J. S. Hoo, "How much is enough? a risk-management approach to computer security," Ph.D. dissertation, 2000. 56

[36] M. A. Sasse, D. Ashenden, L. D., L. Coles-Kemp, F. I., and P. Kearney, "Human vulerabilities in security systems," Tech. Rep., 2007. 56

[37] R. Bojanc and B. Jerman-Blaič, "An economic modelling approach to information security risk management," *Int. J. Inf. Manag.*, vol. 28, no. 5, pp. 413–422, Oct. 2008. 56, 70, 71

[38] A. Bryman, *Social Research Methods*. Oxford University Press, 2001. 57

[39] H. Coombes, *Research using IT*. New York: PALGRAVE, 2001. 57

[40] R. L. Baskerville, "Investigating information systems with action research," *Communications of the Association for Information Systems*, vol. 2, 1999. 57

[41] R. L. Baskerville and A. T. Wood-Harper, "A critical perspective on action research as a method for information systems research," *Journal of Information Technology*, vol. 11, no. 3, pp. 235–246, 2001. 57

[42] A. Stephanou, "The impact of information security awareness training on information security behavior," Tech. Rep., 2008. 57

[43] T. H. Davenport, *Process Innovation: Reengineering work through information technology.* Boston: Harvard Business School Press, 1993. 57

[44] R. S. Aguilar-Saven, "Business process modelling: Review and framework," *International Journal of Production Economics*, vol. 90, no. 2, pp. 129–149, 2004. 57

[45] A. Jaquith, *Security Metrics: Replacing fear, uncertainty and doubt.* Boston: Pearson Education. Inc., 2007. 57, 58

[46] N. Bartol, B. Bates, K. M. Goertzel, and T. Winograd, "Measuring cyber security and information assurance," Information Assurance Technology Analysis Center, Tech. Rep., 2009. 58

[47] H. F. Campbell and R. P. C. Brown, *Benefit-cost analysis.* Cambridge: Cambridge University Press, 2003. 58

[48] J. Umeh, *The World Beyond Digital Rights Management.* The British Computer Society, 2007. 58

[49] O. Pitkanen and M. Valimaki, "Towards a digital rights management framework," in *IeC2000 Proceedings.* 58

[50] E. Becker, W. Buhse, D. Gunnewig, and N. Rump, *Digital Rights Management: Technological, economic, legal and political aspects.* Berlin: Springer, 2003. 58

[51] S. R. Subramanya and B. K. Yi, "Digital rights management," *IEEE Potentials*, vol. 25, no. 2, pp. 31–34, 2006. 58

[52] M. Corporation, "Deploying windows rights management services at microsoft," no. November 20, 2010. 59

[53] M. H. van Beek, "Comparison of enterprise digital rights management systems." Ph.D. dissertation, 2007. 59

[54] W. Zeng and A. van Moorsel, "Quantitative evaluation of enterprise drm technology," *Electron. Notes Theor. Comput. Sci.*, vol. 275, pp. 159–174, Sep. 2011. 64

[55] W. H. Sanders, "Möbius user manual," 2012. 68, 97, 103, 117

[56] A. Consulting., "Specialist and junior information security professions see salary increase," 2010. 71

[57] S. E. Parkin, R. Y. Kassab, and A. van Moorsel, "The impact of unavailability on the effectiveness of enterprise information security technologies," in *Proceedings of the 5th international conference on Service availability*, ser. ISAS'08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 43–58. 75, 78, 93, 121

[58] J. Andrew, "Identification of a method for the calculation of thread in an information environment," in *Internal publication.* QinetiQ, Inc., April 2002. 76, 79

[59] P. R. Stephenson, "A formal model for information risk analysis using colored petri nets," in *Proceedings of the Fifth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, Aarhus, Denmark.* Citeseer, 2004, pp. 167–184. 76, 120

[60] D. D. Deavours and W. H. Sanders, "Möbius: Framework and atomic models," in *Proceedings of the 9th International Workshop on Petri Nets and Performance Models (PNPM'01)*, ser. PNPM '01, 2001, pp. 251–260. 103

[61] Y. Zhao and N. Thomas, "A cost model analysis of a secure key distribution centre," in *ICYCS*, 2008, pp. 1969–1974. 106

[62] E. E. Anderson and J. Choobineh, "Enterprise information security strategies," *Computers & Security*, vol. 27, no. 1-2, pp. 22 – 29, 2008. 109

[63] ADB, "Sensistivity and risk analysis," in *Handbook for the Economic Analysis of Water Supply Projects.* Asian Development Bank, 1999. 109

[64] Scully-Capital, "Scenarios and sensitivity analysis business case for early orders of new nuclear reactors," in *Business Case for Early Orders of New Nuclear Reactors*, 2002. 109

[65] EUcommission, "Commission impact assessment guidelines," in *European Commission Guidelines*, 2009.

[66] D. J. Pannell, "Sensitivity analysis of normative economic models: Theoretical framework and practical strategies," *Agricultural Economics*, vol. 16, pp. 139–152, 1997. 110

[67] J. Dean and L. A. Barroso, "The tail at scale," *Commun. ACM*, vol. 56, no. 2, pp. 74–80, Feb. 2013. 113

[68] N. University, "Login name and password policy in newcastle university," June 2011. [Online]. Available: www.ncl.ac.uk 114

[69] P. G. Inglesant and M. A. Sasse, "The true cost of unusable password policies: password use in the wild," in *Proceedings of the 28th international conference on Human factors in computing systems*, ser. CHI '10. New York, NY, USA: ACM, 2010, pp. 383–392. 114

[70] S. Riley, "Password security: what users know and what they actually do," *Usability News*, vol. 8, 2006. 114

[71] M. A. Sasse, S. Brostoff, and D. Weirich, "Transforming the 'weakest link' - a human computer interaction approach to usable and effective security," *BT Technology Journal*, vol. 19, no. 3, pp. 122–131, Jul. 2001. 114

[72] S. Brostoff and M. A. Sasse, "Ten strikes and you're out: Increasing the number of login attempts can improve password usability," in *Proceedings of CHI 2003 Workshop on HCI and Security Systems*. John Wiley, 2003. 115

[73] C. A. Fidas, A. G. Voyiatzis, and N. M. Avouris, "When security meets usability: A user-centric approach on a crossroads priority problem," *2010 14th Panhellenic Conference on Informatics*, pp. 112 – 117, 2010. 116

[74] J. Zhang, X. Luo, S. Akkaladevi, and J. Ziegelmayer, "Improving multiple-password recall: an empirical study," *European Journal of Information Systems*, vol. 18, no. 2, pp. 165–176, 2009. 116

[75] K. Knorr, "Multilevel security and information flow in petri net workflows," In Proceedings of the 9th International Conference on Telecommunication Systems - Modeling and Analysis, Special Session on Security Aspects of Telecommunication Systems, Tech. Rep., 2001. 118, 120

[76] V. Varadharajan, "Hook-up property for information flow secure nets," in *Computer Security Foundations Workshop IV, 1991. Proceedings*, 1991, pp. 154–175. 118, 120

[77] K. Juszczyszyn, "Verifying enterprise's mandatory access control policies with coloured petri nets," in *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on*, 2003, pp. 184–189. 119, 120

[78] D. E. Robling Denning, *Cryptography and data security*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1982. 119

[79] J. K. Millen, "Covert channel capacity," in *IEEE Symposium on Security and Privacy*, 1987, pp. 60–66. 119

[80] A. McIver and C. Morgan, "Programming methodology," A. McIver and C. Morgan, Eds. New York, NY, USA: Springer-Verlag New York, Inc., 2003, ch. A probabilistic approach to information hiding, pp. 441–460. 119

[81] D. Volpano, C. Irvine, and G. Smith, "A sound type system for secure flow analysis," *J. Comput. Secur.*, vol. 4, pp. 167–187, January 1996. 119

[82] D. Clark, S. Hunt, and P. Malacaria, "Quantitative analysis of the leakage of confidential data," *Electronic Notes in Theoretical Computer Science*, vol. 59, no. 3, pp. 238 – 251, 2002. 119

[83] ——, "Quantified interference for a while language," *Electron. Notes Theor. Comput. Sci.*, vol. 112, pp. 149–166, January 2005. 119

[84] ——, "Quantitative information flow, relations and polymorphic types," *J. Log. and Comput.*, vol. 15, pp. 181–199, April 2005. 119

[85] ——, "A static analysis for quantifying information flow in a simple imperative language," *J. Comput. Secur.*, vol. 15, pp. 321–371, August 2007. 119

[86] M. Andrs, C. Palamidessi, P. van Rossum, and G. Smith, "Computing the leakage of information-hiding systems," in *Tools and Algorithms for the Construction and Analysis of Systems*, ser. Lecture Notes in Computer Science, J. Esparza and R. Majumdar, Eds. Springer Berlin / Heidelberg, 2010, vol. 6015, pp. 373–389. 119

[87] M. E. Andrs, C. Palamidessi, P. van Rossum, and A. Sokolova, "Information hiding in probabilistic concurrent systems." in *QEST'10*, 2010, pp. 17–26. 119

[88] T. Ahmed and A. R. Tripathi, "Static verification of security requirements in role based cscw systems," in *Proceedings of the eighth ACM symposium on Access control models and technologies*, ser. SACMAT '03. New York, NY, USA: ACM, 2003, pp. 196–203. 120

[89] K. Knorr, "Dynamic access control through petri net workflows," in *Computer Security Applications, 2000. ACSAC '00. 16th Annual Conference*, 2000, pp. 159–167. 120

[90] B. Shafiq, A. Masood, J. Joshi, and A. Ghafoor, "A role-based access control policy verification framework for real-time systems," in *in Proc. of IEEE Workshop on Object-oriented Real-time Databases, 2005*, 2005, pp. 13–20. 120

[91] H. Rakkay and H. Boucheneb, "Transactions on computational science iv," M. L. Gavrilova, C. J. Tan, and E. D. Moreno, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, ch. Security Analysis of Role Based Access Control Models Using Colored Petri Nets and CPNtools, pp. 149–176. 120

[92] V. Atluri and W. kuang Huang, "An authorization model for workflows," in *In Proceedings of the 4th European Symposium on Research in Computer Security*. Springer-Verlag, 1996, pp. 44–64. 120

[93] Y. Jiang, C. Lin, H. Yin, and Z. Tan, "Security analysis of mandatory access control model," in *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, vol. 6, 2004, pp. 5013–5018 vol.6. 120

[94] M. Collinson, B. Monahan, and D. J. Pym, "A logical and computational theory of located resource," *J. Log. Comput.*, vol. 19, no. 6, pp. 1207–1244, 2009. 120

[95] Y. Beres, J. Griffin, S. Shiu, M. Heitman, D. Markle, and P. Ventura, "Analysing the performance of security solutions to reduce vulnerability exposure window," in *Proceedings of the 2008 Annual Computer Security Applications Conference*, ser. ACSAC '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 33–42. 121

[96] A. Beautement, R. Coles, J. Griffin, C. Ioannidis, B. Monahan, D. Pym, A. Sasse, and M. Wonham, "Modelling the human and technological costs and benefits of usb memory stick security," in *Managing information risk and the economics of security*, M. E. Johnson, Ed. Boston, MA: Springer US, 2009, pp. 141–163. 121

[97] B. Adrian, M. C. Mont, D. Pym, and S. Shiu, "System modelling for economic analysis of security investments: A case study in identity and access management," in *HPL*, 2009, p. 173. 121

[98] Y. Beres, D. Pym, and S. Shiu, "Decision support for systems security investment," *Manuscript, HP Labs*, 2010. 121