Department of Computing and Software

Faculty of Engineering — McMaster University

Information Leakage via Protocol-Based Covert Channels: Detection, Automation, and Applications

by

Jason Jaskolka, Ridha Khedri, and Khair Eddin Sabri

CAS Report Series **CAS-11-05-RK**

Department of Computing and Software **August 2011**
Information Technology Building
McMaster University
1280 Main Street West Hamilton, Ontario, Canada L8S 4K1

Information Leakage via Protocol-Based Covert Channels: Detection, Automation, and Applications

Jason Jaskolka[1], Ridha Khedri[1], and Khair Eddin Sabri[2]

[1] Department of Computing and Software, Faculty of Engineering, McMaster University, Hamilton, Ontario, Canada

[2] Department of Computer Science, King Abdullah II School for Information Technology, University of Jordan, Amman, Jordan

Technical Report CAS-11-05-RK
Department of Computing and Software
McMaster University

August 23, 2011

Abstract

With the emergence of computers in every day activities and with the ever-growing complexity of networks and network communication protocols, covert channels are becoming an eminent threat to the confidentiality of information. In light of this threat, we propose a technique to detect confidential information leakage via covert channels. Although several works examine covert channel detection and analysis from the perspective of information theory by analysing channel capacities, for instance, we propose a different technique from a different perspective. The proposed technique is based on relational algebra. It provides tests to verify the existence of a leakage of information via a monitored covert channel. The technique also provides computations which, when a leakage is detected, shows how the information was leaked. We also report on a prototype tool that allows for the automation of the proposed technique.

We limit our focus to protocol-based covert channels and instances where the users of covert channels modulate the information that is being sent; either by encryption, or some other form of encoding. We discuss possible applications of the proposed technique in digital forensics and cryptanalysis.

Keywords: covert channel, confidentiality, formal methods, digital forensics, cryptanalysis, security

Contents

1 Introduction and Motivation

With the ever-growing popularity and sophistication of computer systems, computer and information security is becoming more important than ever. Computers are being used in virtually every workplace in some form or another. Hence, due to the widespread use of computers and the variety of application domains, security concerns have varying implications and priority from one domain to another.

Information security has three facets: confidentiality, integrity, and availability [2]. Confidentiality refers to the concealment of information or resources. The demand to keep information concealed arises from the use of computers in government, medical, and industry domains. For example, military institutions restrict access to information to those individuals or groups who have a need for that information. Integrity refers to the trustworthiness of data or resources, and it is usually phrased in terms of preventing improper or unauthorised change. Integrity includes both data integrity, the content of the information, and origin integrity, the source of the data. Origin integrity is commonly referred to as authentication. The accuracy and credibility of information relies heavily on the integrity of information and is central to the proper functioning of a system. Availability refers to the ability to use the information or resource desired. Availability is directly related to the reliability of a system since a system that is unavailable is at least as bad as no system at all. In terms of computer and information security, availability has implications that extend to the ability of an agent to deliberately deny access to data or a service by making it unavailable, thus rendering the system unusable.

In order to discuss information confidentiality, we must have a means of specifying what is, and what is not, a violation of security. Hence, we require a security policy to state what is, and what is not, allowed. According to [38], a security policy is a predicate on the knowledge of a set of agents that establishes what each agent should know and communicate. In [38], an agent's knowledge is captured by a mathematical structure called an Information Algebra. Consider a data store of student records which contains information classified as *name*, *identification number*, and *courses*. A policy for this example may be that no agent should know both the *name* and *identification number* of a student. Confidential information is defined as the information that is protected by the security policy. For the example of a data store of student records, the confidential information can be the combined knowledge of both the *name* and *identification number* of an individual student.

In the area of computer and information security, there are a number of concerns including the leak of confidential information, the unauthorised manipulation of sensitive data, and the denial of a required service. This paper focusses on confidential information leakage via covert channel communication. According to [44], a covert channel is any communication means that can be exploited to transfer information in a manner that violates the system's security policy. We choose to adopt this definition based on its generality and its relationship to the security policy employed by a given system. In [21], the reader finds a thorough survey of covert channels as well as a model for their morphology.

The three facets of information security have a strong relationship to covert channel communication. First, integrity can be compromised by covert channels since covert channeling techniques enable tampering with data stores in a manner that is unknown to the system. Second, covert channels hinder availability since they are able to use system resources to such an extent that it degrades the system's performance and jeopardises its availability. Lastly, covert channels can be used to transmit sensitive information in a secret manner. This makes them

a particular threat to the confidentiality of a system. In modern organisations, the prospect of confidential information leakage ranks among the highest fears of any executive [40]. As many organisations depend on broad and heterogeneous communication networks, the possibilities for the exfiltration of sensitive private information are numerous and the detection of such an event is a challenging problem. For instance, as organisations are beginning to store enormous amounts of data in the "cloud", they must ensure that the cloud is secure. In order to maintain confidentiality, organisations ought to use detection and prevention mechanisms to protect their data and secrets from any sort of attack or leakage. Furthermore, covert channel communication gives rise to economical concerns since they allow for information to be transmitted using an existing system without paying for the service provided. This is often the case when a system is infected by a Trojan Horse. Due to these concerns, among others, covert channel analysis has become part of the evaluation criteria for the classification of secure systems by the United States Department of Defense (DoD) and the National Computer Security Center (NCSC) as outlined in [44] and [46].

According to the United States Department of Homeland Security, there are shortcomings in the science, mathematics, and fundamental theory to deal with covert channels in modern computer systems [45]. We aim to propose mathematical formulations for covert channels and to develop a formal theory for the detection of the leak of confidential information in protocol-based covert channels using algebraic techniques. We do not rule out the possibility that someone may develop heuristics to discover the use of covert channels of various types. However, we are looking to provide a mathematical method which gives a more formal and rigorous approach to uncovering the use of covert channels. A mathematical method for detecting the use of covert channels gives us more power and flexibility than that which could be done with heuristics. It also gives us a significant advantage in that we are able to mechanise and automate the computations needed to discover the use of covert channels and to build and configure monitors which are able to supervise a system for which we strive for confidentiality.

To the best of our knowledge, a formal method such as the one we propose is non-existent. Several works examine covert channel detection and analysis from the perspective of information theory (e.g., [12]) by, for instance, analysing channel capacities. We propose a different technique from a different perspective. This leads to fertile grounds for developing a theory of covert channels and provides us with new and innovative ways to approach the problem of covert channels being a threat to the confidentiality of information. We aim to aid in limiting the nuisance of covert channels which threaten the privacy and confidentiality of information.

In Section 2, we introduce the required mathematical background including sets, relations, and their operations. In Section 3, we describe the process by which we formulate a new technique to detect the leak of confidential information through covert channels. In Section 4, we examine the automation of the proposed technique using a prototype tool implemented in the functional programming language *Haskell*. In Section 5, we discuss the application of the proposed technique in the area of cryptanalysis. In Section 6, we survey the literature and look at existing techniques for mitigating covert channel use. In Section 7, we provide a brief discussion of the proposed technique. Finally, Section 8 draws conclusions and suggests future work.

2 Mathematical Background

In this section, we introduce the necessary mathematical concepts required for the rest of the paper.

2.1 Sets

We aim to detect the use of covert channels through the communication of agents in a system. We use relations to represent the stream of messages sent between agents in a system. Therefore, since relations are defined generally in terms of sets, we first give a general introduction to set theory. The material regarding set theory is extracted from [10].

Definition 1. *A set is a collection of distinct elements.*

There are two ways to describe a set. The first is called set enumeration, which describes a set by listing its elements. For example, $\{1, 8, 27, 64\}$ denotes the set consisting of the elements 1, 8, 27, and 64. The second is called set comprehension, which describes a set by stating properties shared by its elements. For example, the set comprehension $\{x \in \mathbb{N} \mid \exists(y \mid y \in \mathbb{N} \land 1 \leq y \leq 4 : x = y^3)\}$ denotes the set of all natural numbers x such that $\exists(y \mid y \in \mathbb{N} \land 1 \leq y \leq 4 : x = y^3)$ is satisfied.

We identify two special sets: the universal set and the empty set.

Definition 2.

(i) The universal set, denoted by U, is the set fixed within the framework of a theory and consisting of all objects considered in this theory.

(ii) The set $\emptyset$ is called the empty set and is defined by

$$\emptyset \stackrel{def}{=} \{x \mid \mathsf{false}\}$$

The following are a selection of useful operations on sets.

Definition 3. *Let X and Y be sets and let U be the universal set.*

(i) Subset: $X \subseteq Y \iff \forall(x \mid x \in X : x \in Y)$

(ii) Complement: $x \in \overline{X} \iff x \in \mathsf{U} \land x \notin X$

(iii) Union: $x \in X \cup Y \iff x \in X \lor x \in Y$

(iv) Intersection: $x \in X \cap Y \iff x \in X \land x \in Y$

2.2 Relations and Their Operations

Definition 4. *Given two sets, X and Y, we define the Cartesian product $X \times Y$ as*

$$X \times Y = \{(x, y) \mid x \in X \land y \in Y\}$$

Definition 5 (e.g., [39])**.** *Let X and Y be two sets. A relation R on $X \times Y$ is a subset of the Cartesian product $X \times Y$, that is, $R \subseteq X \times Y$. When $X = Y$ we say that R is a homogenous relation and when $X \neq Y$ we say that R is a heterogeneous relation.*

We identify three special relations: the identity relation, the universal relation, and the empty relation.

Definition 6.

(i) For every set X, the relation $\mathbb{I}$ on $X \times X$ is called the identity relation on X and is defined by

$$\mathbb{I} \stackrel{def}{=} \{(x, x) \mid x \in X\}$$

(ii) For every two sets X and Y, the relation $\mathbb{L}$ on $X \times Y$ is called the universal relation and is defined by

$$\mathbb{L} \stackrel{def}{=} \{(x, y) \mid \mathit{true}\}$$

(iii) For every two sets X and Y, the relation $\emptyset$ is called the empty relation and is defined by

$$\emptyset \stackrel{def}{=} \{(x, y) \mid \mathit{false}\}$$

Definition 7 (e.g., [39])**.** *Let $R \subseteq X \times Y$ be a relation:*

(i) The domain of the relation R is given by

$$\mathrm{dom}(R) \stackrel{def}{=} \{x \mid \exists(y \mid y \in Y : (x, y) \in R)\}$$

(ii) The range of the relation R is given by

$$\mathrm{ran}(R) \stackrel{def}{=} \{y \mid \exists(x \mid x \in X : (x, y) \in R)\}$$

There are three important operations on relations that are needed for the rest of the paper: composition, converse and complement.

Definition 8 (e.g., [39])**.**

(i) Let $R \subseteq X \times Y$ and $S \subseteq Y \times Z$ be relations. Then, their composition $R ; S$ is defined by

$$R ; S \stackrel{def}{=} \{(x, z) \mid \exists(y \mid y \in Y : (x, y) \in R \wedge (y, z) \in S)\}$$

(ii) We define the converse of a relation R by

$$R^{\smile} \stackrel{def}{=} \{(x, y) \mid (y, x) \in R\}$$

(iii) We define the complement of a relation R by

$$\overline{R} \stackrel{def}{=} \{(x, y) \mid (x, y) \notin R\}$$

The next definition introduces total, univalent, surjective, injective, mapping, and bijective relations.

Definition 9 (e.g., [39]). *If R is a relation, then we say*

(i) R *is total*
$\iff \mathbb{L} = R;\mathbb{L} \iff \mathbb{I} \subseteq R;R^{\smile}$
$\iff \forall(S \mid\!: R;\overline{S} \supseteq \overline{R;S})$

(ii) R *is univalent (deterministic or functional)*
$\iff R^{\smile};R \subseteq \mathbb{I}$
$\iff \forall(S \mid\!: R;\overline{S} \subseteq \overline{R;S})$

(iii) R *is surjective*
$\iff R^{\smile}$ *is total*

(iv) R *is injective*
$\iff R^{\smile}$ *is univalent*

(v) R *is a mapping*
$\iff R$ *is total and univalent*
$\iff \forall(S \mid\!: R;\overline{S} = \overline{R;S})$

(vi) R *is bijective*
$\iff R$ *is surjective and injective*

Below, we give some important properties of relations from [39].

Proposition 1. *Let P and Q be relations.*

(i) $\overline{\overline{P}} = P$

(ii) $P^{\smile\smile} = P$

(iii) $(P \cup Q)^{\smile} = P^{\smile} \cup Q^{\smile}$

(iv) $(P \cap Q)^{\smile} = P^{\smile} \cap Q^{\smile}$

(v) $(P;Q)^{\smile} = Q^{\smile};P^{\smile}$

The interplay between relational composition, converse, and complement with respect to containment is given by the Schröder equivalences.

Proposition 2. *Let P, Q and R be relations. Then,*

$$P;Q \subseteq R \iff P^{\smile};\overline{R} \subseteq \overline{Q} \iff \overline{R};Q^{\smile} \subseteq \overline{P}$$

Proof. The proof can be found in [39]. □

Definition 10 ([19]). *For every set $A \subseteq X$ and every relation $R \subseteq X \times Y$, we define a relation $R|_A \subseteq X \times Y$, which is the restriction of R to A as*

$$\forall\big(x, y \mid x \in X \wedge y \in Y : (x,y) \in R|_A \iff x \in A \wedge (x,y) \in R\big)$$

Stated otherwise, if we define a predicate $P_A(x)$ which describes the set A, i.e., $x \in A \iff P_A(x)$, and we have a relational expression $R(x, y)$ that defines R, i.e., $(x, y) \in R \iff R(x, y)$, then the restriction of the relation R to A, $R|_A$ is described by the expression $P_A(x) \wedge R(x, y)$. Consider the following example. Let $R = \{(1, a), (2, b), (3, c), (4, d), (5, e), (6, f), (7, g), (8, h), (9, i), (10, j)\}$ and let $A = \{x \mid \mathsf{odd}(x)\}$. Then, $R|_A = \{(1, a), (3, c), (5, e), (7, g), (9, i)\}$.

We introduce the notion of residue which is a special operation on relations. It helps solve equations of the form $P ; X = Q$ or $X ; P = Q$.

Definition 11 ([39]). *Let P and Q be two relations.*

(i) $Q/P \stackrel{def}{=} \overline{\overline{Q} ; P^{\smile}}$ *is called the left residue of Q by P*

(ii) $P \backslash Q \stackrel{def}{=} \overline{P^{\smile} ; \overline{Q}}$ *is called the right residue of Q by P*

The left residue and the right residue are also called, in [16, 17], *weakest prespecification* and *weakest postspecification*, respectively. As an example, let $A = \{a, b\}$ and let $P \subseteq A \times A$ and $Q \subseteq A \times A$ such that $P = \{(a, a), (b, a)\}$ and $Q = \{(a, b), (b, b)\}$. In this case, the universe of values is given as $A \times A$.

We first compute Q/P.

$$
\begin{array}{ll}
 & Q/P \\
= & \quad \langle \text{ Definition 11(i) } \rangle \\
 & \overline{\overline{Q} ; P^{\smile}} \\
= & \quad \langle \text{ Substitution: } P = \{(a, a), (b, a)\} \text{ and } Q = \{(a, b), (b, b)\} \rangle \\
 & \overline{\overline{\{(a, b), (b, b)\}} ; \{(a, a), (b, a)\}^{\smile}} \\
= & \quad \langle \text{ Definition 8(ii) \& Definition 8(iii) } \rangle \\
 & \overline{\{(a, a), (b, a)\} ; \{(a, a), (a, b)\}} \\
= & \quad \langle \text{ Definition 8(i) } \rangle \\
 & \overline{\{(a, a), (a, b), (b, a), (b, b)\}} \\
= & \quad \langle \text{ Definition 8(iii) } \rangle \\
 & \emptyset
\end{array}
$$

Next, we compute $P \backslash Q$.

$$
\begin{array}{ll}
 & P \backslash Q \\
= & \quad \langle \text{ Definition 11(ii) } \rangle \\
 & \overline{P^{\smile} ; \overline{Q}} \\
= & \quad \langle \text{ Substitution: } P = \{(a, a), (b, a)\} \text{ and } Q = \{(a, b), (b, b)\} \rangle \\
 & \overline{\{(a, a), (b, a)\}^{\smile} ; \overline{\{(a, b), (b, b)\}}} \\
= & \quad \langle \text{ Definition 8(ii) \& Definition 8(iii) } \rangle \\
 & \overline{\{(a, a), (a, b)\} ; \{(a, a), (b, a)\}} \\
= & \quad \langle \text{ Definition 8(i) } \rangle \\
 & \overline{\{(a, a)\}} \\
= & \quad \langle \text{ Definition 8(iii) } \rangle \\
 & \{(a, b), (b, a), (b, b)\}
\end{array}
$$

The left residue gives the greatest solution to $X ; P \subseteq Q$ (see Proposition 3(i)). A solution to $X ; P \subseteq Q$ is any relation X such that the equation is satisfied. If the equation $X ; P = Q$ has a solution (i.e., when $\text{ran}(P) = \text{ran}(Q)$), the left residue is its greatest solution. We illustrate this using Figure 1 where P/Q, P, and Q are presented as graphs. We can see that $\text{ran}(P) \cap \text{ran}(Q) = \emptyset$ since $\text{ran}(P) = \{a\}$ and $\text{ran}(Q) = \{b\}$. Therefore, we cannot find a relation $X \neq \emptyset$ such that $X ; P = Q$. Only $X = \emptyset$ satisfies $X ; P \subseteq Q$.

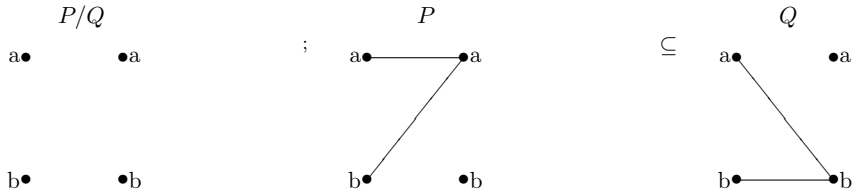


Figure 1: $P/Q ; P \subseteq Q$ with $P/Q = \emptyset$

The right residue is the greatest solution to $P ; X \subseteq Q$ (see Proposition 3(ii)). If $P ; X = Q$ has a solution (i.e., when $\text{dom}(P) = \text{dom}(Q)$), the right residue is its greatest solution. We illustrate this in Figure 2 where P, $P\backslash Q$, and Q are presented as graphs. From these graphs, we can see that $P ; P\backslash Q = Q$. So $P\backslash Q$ is the solution of $P ; X = Q$, i.e., $X = P\backslash Q$.

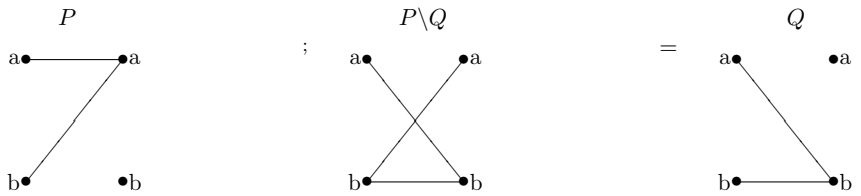


Figure 2: $P ; P\backslash Q = Q$

Proposition 3. *Let P, Q and X be relations.*

(i) $X ; P \subseteq Q \iff X \subseteq Q/P$,

(ii) $P ; X \subseteq Q \iff X \subseteq P\backslash Q$.

Proof. The proof can be found in [25, 39]. □

Some important properties of residues are taken from [6] and are given below.

Proposition 4. *For relations P and Q we have*

(i) $(P/Q)^\smile = Q^\smile \backslash P^\smile$

(ii) $(P\backslash Q)^\smile = Q^\smile / P^\smile$

(iii) $(P/Q) ; Q \subseteq P$

(iv) $Q ; (Q\backslash P) \subseteq P$

In some cases, it is required that a relation be a left residue and right residue simultaneously. This notion is called the symmetric quotient.

Definition 12 ([39]). *If P and Q are relations, we define the symmetric quotient as*

$$\mathrm{syq}(P,Q) \stackrel{def}{=} \overline{P^\smile ; \overline{Q}} \cap \overline{\overline{P^\smile} ; Q} = (P\backslash Q) \cap (P^\smile / Q^\smile)$$

The symmetric quotient $\mathrm{syq}(P,Q)$ of two relations P and Q is defined as the greatest relation X such that $P ; X \subseteq Q$ and $X ; Q^\smile \subseteq P^\smile$. For example, let $A = \{a, b\}$, and $P \subseteq A \times A$ and $Q \subseteq A \times A$ such that $P = \{(a,a),(b,a)\}$ and $Q = \{(a,b),(b,b)\}$. In this case, the universe of values is $A \times A$.

We compute $\mathrm{syq}(P,Q)$.

$$\begin{array}{ll}
 & \mathrm{syq}(P,Q) \\
= & \quad \langle \text{ Definition 12 } \rangle \\
 & \overline{P^\smile ; \overline{Q}} \cap \overline{\overline{P^\smile} ; Q} \\
= & \quad \langle \text{ Substitution: } P = \{(a,a),(b,a)\} \text{ and } Q = \{(a,b),(b,b)\} \rangle \\
 & \overline{\{(a,a),(b,a)\}^\smile ; \overline{\{(a,b),(b,b)\}}} \cap \overline{\overline{\{(a,a),(b,a)\}^\smile} ; \{(a,b),(b,b)\}} \\
= & \quad \langle \text{ Definition 8(ii) \& Definition 8(iii) } \rangle \\
 & \overline{\{(a,a),(a,b)\} ; \{(a,a),(b,a)\}} \cap \overline{\{(b,a),(b,b)\} ; \{(a,b),(b,b)\}} \\
= & \quad \langle \text{ Definition 8(i) } \rangle \\
 & \overline{\{(a,a)\}} \cap \overline{\{(b,b)\}} \\
= & \quad \langle \text{ Definition 8(iii) } \rangle \\
 & \{(a,b),(b,a),(b,b)\} \cap \{(a,a),(a,b),(b,a)\} \\
= & \quad \langle \text{ Definition 3(iv) } \rangle \\
 & \{(a,b),(b,a)\}
\end{array}$$

Some important results used in the remainder of this paper regarding the properties of relations and residues are given in Proposition 5.

Proposition 5. *Let P and Q be relations.*

(i) P is a bijection $\wedge$ Q is surjective $\implies$ $P\backslash Q$ is surjective

(ii) $P\backslash Q = (Q\backslash P)^\smile \implies P \subseteq Q ; (Q\backslash P)$ for Q a bijection and P surjective

(iii) $P \subseteq Q ; (Q\backslash P) \wedge Q \subseteq P ; (P\backslash Q) \iff P\backslash Q = (Q\backslash P)^\smile$ for P and Q bijections

Proof. The detailed proof can be found in Appendix A.1. The proof for (i) involves the properties of total and surjective relations, as well as the application of Definition 11 and Proposition 1. In the proof for (ii), we use the complement and converse properties of relations and apply Definition 11, Proposition 1, Proposition 3, and Proposition 4. The proof for (iii) involves the properties of total and surjective relations and the properties of residues. We apply Definition 11, Proposition 1 and Proposition 4. □

3 Formulation of a Detection Technique

In this section, we formulate our proposed technique for detecting confidential information leakage via covert channels in computer and information systems.

3.1 Assumptions

In formulating the problem of covert channel communication in computer and information systems, we make some basic assumptions regarding the problem in order to simplify the proposed detection technique. Firstly, it is assumed that the communicating agents have a predefined scheme regarding how the information is transmitted from its source to its destination. This includes an agreement on the protocol to be exploited and the fields of the data structure to be used. This is a common assumption among the literature [4, 14, 24, 32]. This assumption is required in order to ensure that the receiving agent is able to recover the communicated information. Next, it is assumed that the communication among agents is recorded by monitors which begin recording when a communication channel is established. It is also assumed that the monitors maintain an unbounded history of all of the communication which has taken place. With these assumptions, we are able to ensure that the monitors have a record of all of the transmissions between the communicating agents. This allows for simplicity in reasoning about the abilities of the system monitors. A similar assumption is made in [7, 11, 14, 50], where the monitors or wardens have access to all of the messages passed between the communicating agents. Another basic assumption is that the monitor always knows the set of confidential information that is protected by the security policy. Lastly, the analysis is done in a forensics context, meaning that it is performed after the information has already been sent. In [35, 42], we find that it is common for an organisation to gather and preserve digital evidence such as transaction logs before an incident occurs. This way, when there is reason to suspect that some violation of the security policy may have been committed using a computer, either in a stand alone manner or in a network environment, analyses can be performed.

3.2 Representing Covert Channels as Relations

Finding an appropriate abstract representation for the information being sent on a channel is a crucial step in solving the problem of detecting the use of covert channels to leak confidential information. Without an appropriate representation for information, we are unable to accurately model the scenarios in which confidential information is leaked via a covert channel. In [21], we discussed how we can view information sent over covert channels as being encapsulated in a data structure of some dimension. This data structure has fields in which the information is embedded. In this paper, we represent the information sent on a channel as a relation, i.e., a series of data structures which are sent over time. At each time, we have an element of information sent. Therefore, one can see a stream of information as a subset of the Cartesian product of time and the state space of a data structure. If we model time by $\mathbb{N}$, and the set of information (or data) by $\mathbb{D}$, then a stream S is a subset of $\mathbb{N} \times \mathbb{D}$. Therefore, it is a relation and more precisely, it is a function when we consider only one channel (without noise). We associate each datum with the time stamp at which it was received. For example, if the data sent on the channel was the sequence of characters 'h', 'e', 'l', 'l', 'o' to form the word "hello", the information that is sent on the stream is formed as the relation $R = \{(1, \text{'h'}), (2, \text{'e'}), (3, \text{'l'}), (4, \text{'l'}), (5, \text{'o'})\}$, where 'h' was sent at time 1, 'e' was sent at time 2 and so on.

In order to uncover a confidential information leakage via a covert channel, we show that it is sufficient to find an abstraction relation between the confidential information which we do not want to be leaked and the stream of information observed to be sent on the channel. An

abstraction relation X can be seen as a simulation relation between two relations P and Q. In Figure 3, the relation X is an abstraction relation that relates $\mathbb{D}_P$ to $\mathbb{D}_Q$.

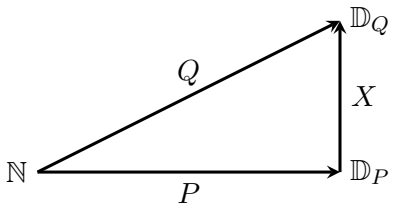
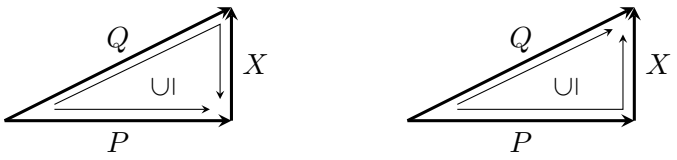


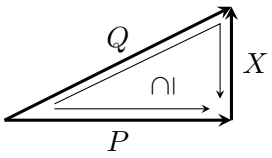
Figure 3: A representation of the relationship between the relations P and Q via the abstraction relation X

In Figure 3, we have the relation P representing the confidential information which should not be sent on the channel, the relation Q representing the information that is observed by a monitor watching the information transmitted over the communication channel, and the relation X representing an abstraction relation that gives the relationship between P and Q. An abstraction relation X, requires that the diagram given in Figure 3 commutes. We can see that the diagram in Figure 4 can commute in four ways as described in Figure 4.

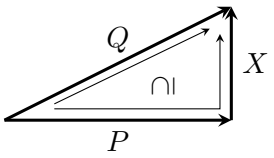


(a) $P \subseteq Q ; X^{\smile}$

(b) $P ; X \subseteq Q$



(c) $Q ; X^{\smile} \subseteq P$



(d) $Q \subseteq P ; X$

Figure 4: Four ways in which the diagram of Figure 3 can commute

However, we can see that the diagrams in Figure 4 can be reduced to two diagrams. For Figure 4a and Figure 4c we have $P \subseteq Q ; X^{\smile} \ \wedge \ Q ; X^{\smile} \subseteq P$ which is equivalent to

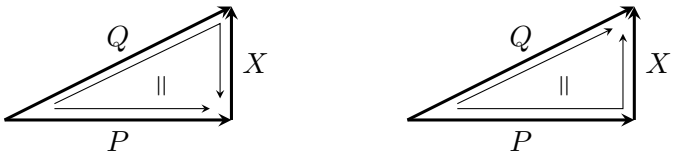
$$Q ; X^{\smile} = P \quad \text{which is equivalent to} \quad X ; Q^{\smile} = P^{\smile} \tag{1}$$

and for Figure 4b and Figure 4d we have $P ; X \subseteq Q \ \wedge \ Q \subseteq P ; X$ which is equivalent to

$$P ; X = Q \quad \text{which is equivalent to} \quad X^{\smile} ; P^{\smile} = Q^{\smile} \tag{2}$$

So, we have the following two diagrams, given in Figure 5, for which we are able to solve their corresponding equations for an abstraction relation X.

In each case, the confidential information represented by P is known. The observed information represented by Q is known only after observing the information that is sent on the



(a) $Q; X^{\smile} = P$ (b) $P; X = Q$

Figure 5: Reduction of the ways in which the diagram of Figure 3 can commute

communication channel. We are looking to find a solution to Equation 1 or Equation 2. The solution is an abstraction relation X relating the relation P and the relation Q. In terms of covert channels, the solution is an abstraction relation relating the confidential information and the observed information that has been sent on the communication channel.

The mathematics required to solve for the abstraction relation X in the diagrams given in Figure 5 was first introduced by Bertrand Russell who wrote on the similarity of relations in 1919 [36]. Russell wrote that two relations are similar when there is at least one abstraction relation between them. When two relations are similar, they share all properties that do not depend upon the actual terms of their fields. This means that if we can find an abstraction relation between the confidential information and the observed information sent on a covert channel, then we can conclude that there is a similarity between the confidential information and the observed information. This means that each element of the confidential information can be mapped to one or more elements of the observed information. This is indeed what we see in Figure 3, whereby composing the relation P with the abstraction relation X, we get the relation Q. Putting this in terms of the communication of information over covert channels, we have that the transformation of the confidential information by the abstraction relation gives the information observed on the communication channel. It is important to emphasise that it is not the communication between agents that is in violation of a security policy, but rather, the information that is being communicated. We are only interested in finding an abstraction relation between the confidential information that is known to the monitor and the observed information sent on the channel(s).

When considering the detection of confidential information leakage via covert channels, we must determine the necessary and sufficient conditions which imply the existence of a covert channel in violation of a security policy. We consider an information leak to be detected if and only if there exists an abstraction relation between the confidential information and the information that is sent on the covert communication channel such that the abstraction relation is different from $\emptyset$ and $\mathbb{L}$. The case where an abstraction relation is equal to $\emptyset$ indicates that there is no abstraction and thus no relationship between the confidential information and the information observed to be sent over the communication channel. The case where an abstraction relation is equal to $\mathbb{L}$ indicates that all information is related to all other information. In this case, the abstraction becomes irrelevant.

3.3 The Proposed Technique

The proposed technique for the detection of confidential information leakage via covert channels has two major components: monitoring the information sent on the communication channel and finding and computing an abstraction relation relating the confidential information to the information observed to be sent on the communication channel.

3.3.1 Illustrative Example

We give a simple example to illustrate the proposed technique in detecting confidential information leakage via covert channels. We use this example as a running example throughout the remainder of this paper.

Consider a system for which two agents are communicating. Suppose that agent A is communicating from within an organisation and suppose that agent B is communicating from outside the organisation. Suppose that the organisation has a security policy which defines its confidential information to be the sequence of the first ten digits of the number π, i.e., we have $P = \{(1,3), (2,1), (3,4), (4,1), (5,5), (6,9), (7,2), (8,6), (9,5), (10,3)\}$ which is a representation of the sequence $\langle 3, 1, 4, 1, 5, 9, 2, 6, 5, 3\rangle$.

Assume that agent A and agent B agree on a scheme for transmitting the confidential information. It is decided that agent A will exploit the Internet Protocol, in particular, the IP *Identification* field in order to leak the confidential information to agent B. The IP *Identification* field is used to uniquely identify an IP datagram within a flow of datagrams that share the same source and destination. Since the value for the IP *Identification* field should be chosen at random, it is possible to choose a non-random value for the field without interrupting the IP mechanism.

Suppose that agent A uses the 16-bit IP *Identification* field to send, in a sequence of IP datagrams, the set of confidential information of its organisation. Also, suppose that in order to attempt to mask the data being sent, agent A first encrypts the information before embedding it into the IP header. For this purpose, agent A uses a public key encryption technique (though it is not important how the information is encrypted) to encrypt the information. The encryption generates the sequence[1] $\langle 12, 1, 16, 1, 17, 18, 11, 6, 17, 12\rangle$ in place of the sequence $\langle 3, 1, 4, 1, 5, 9, 2, 6, 5, 3\rangle$.

Using this particular example for illustration, we are able to show how we can detect confidential information leakage via covert channel communication. Since the example has redundancy in the information, i.e., it contains more than one instance of the digits, it allows for the demonstration of the technique and aids in explaining when the detection of the leakage of confidential information using the proposed technique can be averted. In Section 5.1, we provide a larger scale example through a case study of the Zodiac 408 cipher for which we demonstrate the use of the proposed technique in the context of cryptanalysis.

3.3.2 Monitoring the Communication Channels

Suppose that the organisation from which agent A is communicating wishes to detect if its confidential information is being leaked to an agent outside of the organisation. The organisation installs a monitor on the known communication channels from which an agent from within the organisation can communicate with an agent outside the organisation. The monitor begins monitoring the communication over a channel when the communication channel is established and it keeps a history of the all of the communication that has been observed on the channel, denoted by Q. We can view the monitor as a specialised and customisable packet sniffer in the case of covert channels exploiting the use of network protocols. In order for the monitor to be effective, we must assume that it is configured with the following necessary information:

[1] This set is generated using RSA encryption with $p = 3$, $q = 7$, $N = 21$, $e = 5$, $d = 41$.

- *Protocol*: This refers to the protocol in which the communicating agents are using in order to communicate. In our example, the protocol that is being used is the Internet Protocol (IP).
- *Header Field*: This refers to the field in the header of the particular protocol which is being used as the carrier for the covert information channel. In our example, the field that is being exploited is the IP *Identification* field.
- *Confidential Information*: The monitor must know the confidential information in order to perform the required analysis. In our example, the confidential information that the monitor must know is P.
- *Analysis Tools*: This refers to the set of tests which can be run in order to verify whether there is an abstraction relation between the confidential information and the information that is observed by the monitor. This will be discussed further in Section 3.3.3 and Section 3.3.4.

The monitor of the communication channels that is installed by the organisation can perform either a post-mortem analysis[2] or a real-time analysis looking for an abstraction relation between P and Q.

The monitor watches the stream of packets being transmitted from agent A to agent B. Based on its configuration, the monitor can either extract the header field from the protocol packets that it is monitoring as they are being transmitted and store them, or it can mirror and store the packets of the protocol and then extract the prescribed header field at a later time so as not to interrupt the performance of the communication channel. The monitor finds that agent A sends the confidential sequence of information $\langle 3,\ 1,\ 4,\ 1,\ 5,\ 9,\ 2,\ 6,\ 5,\ 3\rangle$ as the encrypted sequence of information $\langle 12,\ 1,\ 16,\ 1,\ 17,\ 18,\ 11,\ 6,\ 17,\ 12\rangle$, then the relation constructed by the monitor would be given as $Q = \{(1,12), (2,1), (3,16), (4,1), (5,17), (6,18), (7,11), (8,6), (9,17), (10,12)\}$. Recall that the monitor is already equipped with the relation corresponding to the set of confidential information, P. So, the monitor now knows the stream of confidential information which should not be sent according to the system security policy and the stream of information observed to have been sent on the communication channel. The monitor needs to determine whether the confidential information has been leaked in any capacity. It decides if there was a confidential information leakage by verifying the existence of an abstraction relation X which relates the relation constructed as P, the confidential information, and the relation constructed as Q, the observed information.

3.3.3 Finding an Abstraction Relation

In this section, we provide the necessary and sufficient conditions which imply the existence of a covert channel in violation of the system security policy.

Since the monitor knows the relation corresponding to the set of confidential information P and the relation corresponding to the set of observed information Q, the existence of an abstraction relation X which relates P and Q can be verified using Proposition 6.

[2]Post-mortem analysis refers to the fact that the analysis is being done in a digital forensics context whereby confidential information may have already been leaked and the damage may already be done.

Proposition 6. *$X ; P = Q$ has a solution if and only if $Q = (Q/P) ; P$.*

Proof. The detailed proof can be found in Appendix A.2. The proof involves the application of Proposition 3, trading rules, the One-Point Axiom and the isotony of relational composition. □

Proposition 6 is used as a test to verify if there is an abstraction between the observed information and the confidential information. This test is directly related to Figure 3 in that if the test holds, we can say that the diagram in Figure 3 commutes and we can find an abstraction relation X that satisfies Equation 1 or Equation 2 which is not the empty relation $\emptyset$ or the universal relation $\mathbb{L}$. Therefore, we can say that the confidential information P seems to have been leaked using the abstraction given by X and was sent as the observed information Q.

Corollary 1. *Let P be the relation containing confidential information. Let Q be a relation representing an information observed on the monitored communication channel. The confidential information contained in P is being leaked as that represented by Q if and only if*

$$P = Q ; (Q \backslash P) \ \vee \ Q = P ; (P \backslash Q)$$

Proof. The detailed proof can be found in Appendix A.3. The proof involves the application of Proposition 6, Proposition 1(ii), Proposition 1(v) and Proposition 4(i). □

In order for an abstraction relation to exist, we require that the diagram in Figure 3 commutes. The diagram in Figure 3 can commute in two ways as described by Figure 5. In Corollary 1, we verify whether the diagram commutes in at least one of the two ways. Each term of the disjunction in the test corresponds to one of the ways in which Figure 3 can commute. The term $P = Q ; (Q \backslash P)$ corresponds to Figure 5a and the term $Q = P ; (P \backslash Q)$ corresponds to Figure 5b. Therefore, as long as we can satisfy at least one of the ways in which the diagram commutes, we can find an abstraction relation X that satisfies Equation 1 or Equation 2 which is not the empty relation or the universal relation. Then, we can conclude that the confidential information has been leaked via the covert communication channel on which the observed information was sent.

In our example, the confidential information is represented by $P = \{(1,3),\ (2,1),\ (3,4),\ (4,1),\ (5,5),\ (6,9),\ (7,2),\ (8,6),\ (9,5),\ (10,3)\}$ and the information observed by the monitor is represented by $Q = \{(1,12),\ (2,1),\ (3,16),\ (4,1),\ (5,17),\ (6,18),\ (7,11),\ (8,6),\ (9,17),\ (10,12)\}$. Using our prototype tool (see Section 4) to automate the application of Corollary 1, we find that the test holds. We interpret this result as saying that there exists an abstraction relation which relates the confidential information P to the observed information Q meaning that the confidential information has been leaked in some form on the communication channel.

3.3.4 Computing the Abstraction Relation

Now that we have verified whether an abstraction relation does in fact exist, the next step is to compute the abstraction relation giving us the abstraction that is used to relate the confidential information and information that is sent on the channel. Using Proposition 7, we are able to compute the abstraction relation X. The proposition also allows for filtering on the abstraction relation to look for an abstraction which maps particular elements of the confidential information to particular elements of the observed information. The filtering relation is designed by the analyst and is represented by R.

Proposition 7. *Let P and Q be relations. $X ; P = Q$ has a solution $X = R \cap (Q/P)$ if and only if $Q \subseteq \big(R \cap (Q/P)\big) ; P$.*

Proof. The detailed proof can be found in Appendix A.4. The proof involves the application of Definition 11, Proposition 1 and Proposition 4, as well as anti-symmetry and weakening. □

The relation R plays the role of a filter. A filter allows for the removal of some unwanted elements of the transmitted sequences. The filter R allows us to select only those elements of the transmitted sequences which we are interested in examining further. In its most general case, if we consider the filter R to be the universal relation $\mathbb{L}$, we are interested in all of the elements of the transmission. Otherwise, we can select the elements of the range of the confidential information for which we wish to find an abstraction by choosing different filtering relations for R. For instance, if we suspect that the confidential information is sent using only a subset $\mathbb{S}_P$ of $\mathbb{D}_P$, then we can filter using the relation $R = \{(d_p, d_q) \mid d_p \in \mathbb{S}_P \wedge d_q \in \mathbb{D}_Q\}$. By computing an abstraction relation which is not the empty relation $\emptyset$ or the universal relation $\mathbb{L}$, we can say that we have uncovered a leak of confidential information on the communication channel.

Corollary 2. *Let P be the relation containing confidential information. Let Q be a relation representing an information observed on the monitored communication channel. Let R be a filtering relation allowing for the selection of particular elements of the relations P and Q. The confidential information included in P is being leaked as that represented by Q via the abstraction relation X such that*

(i) $X = R \cap (Q \backslash P)^\smile$ *if and only if* $P \subseteq Q ; \big(R^\smile \cap (Q \backslash P)\big)$

(ii) $X = R \cap (P \backslash Q)$ *if and only if* $Q \subseteq P ; \big(R \cap (P \backslash Q)\big)$

(iii) $X = R \cap \mathrm{syq}(P, Q)$ *if and only if* $P \subseteq Q ; \big(R^\smile \cap (Q \backslash P)\big) \wedge Q \subseteq P ; \big(R \cap (P \backslash Q)\big)$

Proof. The detailed proof can be found in Appendix A.5. The proofs for (i), (ii) and (iii) involve the application of Proposition 1 and Proposition 7. The proof for (iii) also involves the Golden Rule Axiom. □

Corollary 2 gives three cases for which we can compute the abstraction relation X. In each of these cases, we compute the abstraction relation X according to the way(s) in which Figure 3 commutes. Corollary 2(i) corresponds to the case where the diagram commutes only as in Figure 5a. Corollary 2(ii) corresponds to the case where the diagram commutes only as in Figure 5b. Lastly, Corollary 2(iii) corresponds to the case where the diagram commutes as in both Figure 5a and Figure 5b.

Continuing with our example, we have already verified the existence of an abstraction relation relating the confidential information P to the observed information Q. Now, using our prototype tool to compute the application of Corollary 2 with the filter $R = \mathbb{L}$, we find that the abstraction relation X contains[3] $X' = \{(1,1), (2,11), (3,12), (4,16), (5,17), (6,6), (9,18)\}$. Here, we have $X' \subseteq \mathbb{D}_P \times \mathbb{D}_Q$. We can interpret X' by reading that a 1 in the confidential information was sent as 1 through the communication channel, that a 2 in the

[3]Here we say X contains X' since the abstraction relation X will have elements relating the unused digits, i.e., $7, 8, 0$, to all elements of the ring of integers modulo N, $\mathbb{Z}/N\mathbb{Z}$ where $N = 21$ comes from the use of RSA for encryption. For brevity, we are simply removing these elements from the relation X to obtain X'.

confidential information was sent as 11 through the communication channel and so on. The abstraction relation X shows how the confidential information P was transformed into the observed information Q.

It is possible when the confidential information and the information observed to be sent on the channel have certain properties, namely if they are bijections, that we can have a specialised case of Corollary 2 where the test is simplified based on the results of Proposition 5. This simplified test and computation is given in Corollary 3.

Corollary 3. *Let P be a bijection containing confidential information. Let Q be a bijection representing an information observed on the monitored communication channel. Let R be a filtering relation allowing for the selection of particular elements of the relations P and Q. The confidential information contained in P is being leaked as that represented by Q via the abstraction relation $X = R \cap (P \backslash Q)$ if and only if $P \backslash Q = (Q \backslash P)^{\smile}$.*

Proof. The detailed proof can be found in Appendix A.6. The proof involves the properties of bijective relations and the application of Proposition 1, Proposition 4, and Proposition 5. □

Examples illustrating Corollary 3 are mainly trivial since they involve P and Q being bijections. Although applications of Corollary 3 may be limited, it is still an important result as it can be seen as a special case for computing the abstraction relation X with simplified conditions.

3.3.5 Modulating the Confidential Information Prior to Transmission

Suppose that agent A and agent B develop a new scheme to mask their covert communication of confidential information. Assume that the agents decide to modulate the confidential information prior to its encryption. This means that they modify the confidential information by some agreed upon scheme and then encrypt the modulated information so as to add another level of abstraction to the transmitted information. Proposition 8 shows how the modulation of the confidential information prior to the encryption and transmission makes no difference on the ability to detect whether it has been leaked in some form. This highlights the point that the encryption of the information does not matter. Since we know the confidential information and we observe the information that is being sent on the channel, we do not need to know how the information was encrypted. If an abstraction relation exists between the confidential information P and the observed information Q, then even if a modulation of P by some relation M is transmitted as Q, we can still find an abstraction relation relating P and Q without knowing M.

Proposition 8. *Let P be a relation containing confidential information. Let M be a total and injective relation that modulates the confidential information in some way such that the modulated confidential information is represented by $(P ; M)$. Let Q be a relation representing an information observed on the monitored communication channel. If the confidential information contained in P is being leaked as that represented by Q then any modulation of the confidential information contained in P is also being leaked as that represented by Q (i.e., $\exists(X \ |: \ P ; X = Q) \implies \exists(Y \ |: \ P ; M ; Y = Q)$).*

Proof. The detailed proof can be found in Appendix A.7. The proof involves the properties of total and injective relations, as well as the application of Proposition 1, Proposition 4, and Proposition 6. □

In the case of modulating the confidential information, we require that the modulation relation be total and injective. The totality of the modulation relation ensures that, when composed with the relation containing the confidential information, no information is lost, i.e., all of the confidential information is represented in some form in the modulated confidential information. The injectivity of the modulation relation ensures that no inconsistencies are introduced which will cause the tests to be averted (see Section 3.3.6). The introduction of an inconsistency in the modulated confidential information leads to a loss of information. The purpose of the conditions on the modulation relation is to ensure that no information is lost during the modulation of the confidential information.

We modify our example from Section 3.3.1 to illustrate the case where the two communicating agents modulate the confidential information prior to its encryption and transmission. Consider now that in order to obscure the transmission of the information, agent A modulates the confidential information by a relation represented by $M = \{(0,9), (1,0), (2,1), (3,2), (4,3), (5,4), (6,5), (7,6), (8,7), (9,8)\}$ prior to its encryption. Agent A computes $(P ; M) = \{(1,2), (2,0), (3,3), (4,0), (5,4), (6,8), (7,1), (8,5), (9,4), (10,2)\}$. Now, using the same encryption scheme as before, agent A encrypts the information and sends it to agent B on a single communication channel as the relation $Q = \{(1,11), (2,0), (3,12), (4,0), (5,16), (6,8), (7,1), (8,17), (9,16), (10,11)\}$. We verify the existence of an abstraction relation by applying Corollary 1. In this case, we are looking for an abstraction relation relating the confidential information P and the observed information Q which corresponds to the encrypted modulated confidential information. Using our prototype tool, we find that an abstraction relation does in fact exist relating the confidential information P and the observed information Q. We then use the prototype tool to apply Corollary 2 with the filter $R = \mathbb{L}$ to compute the abstraction relation X which contains $X' = \{(1,0), (2,1), (3,11), (4,12), (5,16), (6,17), (9,8)\}$. This example illustrates how the modulation of the confidential information prior to the encryption and transmission makes no difference on the ability to detect whether it has been leaked in some form.

3.3.6 Averting the Test for an Abstraction Relation

The test outlined in Proposition 6 can be averted when an element of the confidential information maps to more than one element of the information observed to be sent on the communication channel, of which another element of the confidential information is already mapped. In many cases, the failure of the test means that there is no abstraction between the confidential information and the information observed to be sent on the channel and thus, the confidential information is not being leaked through the communication channel. However, it is possible that an abstraction exists between parts of the confidential information and the information observed to be sent on the channel.

If we take our running example and consider the case where agent A makes an error when representing the confidential information, i.e., instead of the confidential information being $P = \{(1,3), (2,1), (3,4), (\mathbf{4},\mathbf{1}), (5,5), (6,9), (7,2), (8,6), (9,5), (10,3)\}$, it is represented as $P' = \{(1,3), (2,1), (3,4), (\mathbf{4},\mathbf{3}), (5,5), (6,9), (7,2), (8,6), (9,5), (10,3)\}$. Now, assume that this information P' is encrypted and sent to agent B as $Q = \{(1,12), (2,1), (3,16), (\mathbf{4},\mathbf{12}), (5,17), (6,18), (7,11), (8,6), (9,17), (10,12)\}$. We can automatically verify the existence of an abstraction relation using our prototype tool to apply Corollary 1. We find that the result of the test is false. This means that no confidential information has been leaked, which is the case (strictly speaking). Thus, we can conclude that an abstraction relation cannot be

computed and that the confidential information represented by P is not being leaked as that represented by Q. This is due to the inconsistency introduced at times 1, 2, and 4. The data observed at each of these times generates an abstraction relation which cannot be used to accurately uncover the confidential information at those times. This is to say that when we observe a 12 to have been sent on the communication channel, we are unable to determine whether that 12 corresponds to 1 or to 3 in the confidential information. With this example, it is obvious that if the element which was introduced by the error of agent A, i.e., $(4, 3)$, is removed, that an abstraction exists between the confidential information and the information observed to be sent on the channel. Thus, we emphasise that even if the test fails, it is possible for an abstraction to exist between parts of the confidential information and the information observed to be sent on the channel. Currently, we are not able to handle such as scenario and it is left as future work.

The ideas presented in Proposition 6 and Proposition 7, and consequently Corollary 1 and Corollary 2, are the core of the detecting whether confidential information has been leaked via covert channels. Equipped with each of the above propositions and corollaries, and under our assumptions, it is possible to detect the leak of confidential information via monitored covert channels in a digital forensics context, i.e., investigation after the information has already been sent. We do not rule out the possibility that a real-time analysis can be performed, but we do not investigate this issue in this paper. The above propositions and corollaries represent tests for which we can determine the existence of an abstraction relation defining the relationship between the confidential information and the information observed to be sent on the communication channel. The existence of an abstraction relation is often enough to raise suspicion that confidential information has been leaked via a covert channel. Hence, we have formulated a mathematical framework for the post-mortem detection of the leak of confidential information via covert channels.

4 Automation

To automate the process of detecting confidential information leakage via covert channel communication, we implemented a prototype tool, written in the functional programming language *Haskell*. This prototype tool is used to automate the tests and computations presented in Section 3.3.

4.1 Architecture Design

In designing the prototype tool, we opted for a layered architecture. Each layer of the tool specialises in a set of related activities. The architectural design of the prototype tool is given in Figure 6.

The layered architectural style is suitable for our need for a number of reasons. Firstly, a layered architecture enables the incremental development of the tool based on increasing levels of abstraction. For instance, each layer of our tool, with the exception of the *User Interface Layer*, corresponds to a level of abstraction. The layered architecture also lends to enhanced independence among layers. This is to say that there is no impact from changes of lower services provided that their interface to the other layers is maintained. Again, this independence allows for better maintainability and flexibility in the technology used for a given layer. For example, we are able to change the database system that is used in the

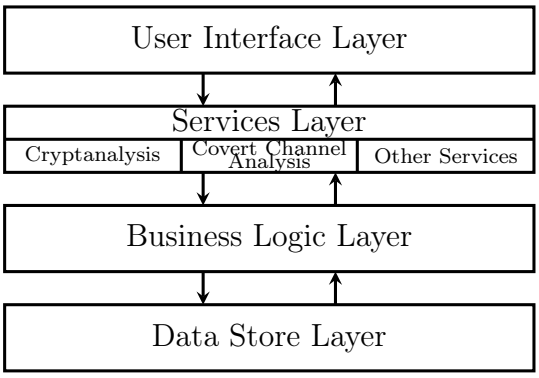


Figure 6: Main layers of the prototype tool's architecture

Data Store Layer without affecting the rest of the system as long as we do not change the interface to the other layers. This is an application of the principle of information hiding and gives us a modular system. Lastly, since the layered architecture is a modular architectural style, it is suitable for plug-and-play components. Indeed, in future work, we will wish to add more services and functionality to the tool and this particular characteristic of the layered architecture lends to a simple integration of new components, particularly at the *Services Layer*.

The *Data Store Layer* is responsible for storing and retrieving information from the file system. The *Data Store Layer* keeps the data independent from the business logic which allows for improved scalability and performance. The prototype tool implements the *Data Store Layer* as a file system consisting of data files used to store the relational representation of the confidential information and the observed information that has been sent on the communication channel.

The *Business Logic Layer* is responsible for coordinating the flow of data from the *Data Store Layer* to the *Services Layer*. This layer handles the parsing of the data files, generating the internal relational representation of the data and relaying the results to the appropriate service offered by the *Services Layer*.

The *Services Layer* currently provides two main services which employ the proposed technique for two different applications:

(i) *Covert Channel Analysis*: The covert channel analysis service offers tools to detect the leak of confidential information through covert channel communication. These tools correspond to the corollaries of the proposed technique from Section 3.3 and can be used to verify whether confidential information has been leaked and to compute the corresponding abstraction relation if it is found that one does indeed exist. An example use of the prototype tool with emphasis on the covert channel analysis service is given in Section 4.2.

(ii) *Cryptanalysis*: The cryptanalysis service offers the ability to perform a cryptanalysis based on the proposed technique and a known-plaintext attack. Further details regarding the application of the proposed technique in cryptanalysis are given in Section 5.

The *User Interface Layer* supports the display and input of service commands. Currently, the prototype tool has a command-line interface which is run through the Glasgow Haskell Compiler's interactive environment (`ghci`). In Section 4.2, the reader finds examples on the usage of the tool.

4.2 Example Tool Use

In this section, we show the use of the prototype tool to automate the detection of confidential information leakage via covert channel communication through the use of our running example.

A monitor extracts the IP *Identification* field from the transmitted IP datagrams associated with the communication between agent A and agent B. The monitor collects this data and stores it in a file for processing, i.e., verification of the existence of a confidential information leakage. Suppose that the information that the monitor records is the sequence $\langle 12, 1, 16, 1, 17, 18, 11, 6, 17, 12\rangle$. Then, this observed information is stored in a file which, from this point forward, we call `observed.seq`.

Assume that the monitor is already configured with the set of confidential information, which in this case is the sequence $\langle 3, 1, 4, 1, 5, 9, 2, 6, 5, 3\rangle$. This sequence is stored in a file which we call `confidential.seq`.

Now, we wish to analyse the information that the monitor observed to have been sent on the communication channel to verify whether the confidential information has been leaked in some form. We start by loading the prototype tool modules in the Glasgow Haskell Compiler's interactive environment (`ghci`) with the following command:

```
> :load PrototypeTool
```

Before we start to process the observed information collected by the monitor, we create a new data store file called `ExampleDB` to store all the relations for this session so that we can quickly recall them later. This is done by issuing the following command:

```
> new "ExampleDB"
```

Next, the files are loaded into the prototype tool to construct the internal relational representation of the information, i.e., the contents of the files `observed.seq` and `confidential.seq` are represented as the relations $\{(1,12), (2,1), (3,16), (4,1), (5,17), (6,18), (7,11), (8,6), (9,17), (10,12)\}$ and $\{(1,3), (2,1), (3,4), (4,1), (5,5), (6,9), (7,2), (8,6), (9,5), (10,3)\}$, respectively. The internal relational representation is stored in our data store file, `ExampleDB`. This is done by issuing the following commands:

```
> loadRel "observed.seq" "observed" "ExampleDB"
> observed <- select "observed" "ExampleDB"
> loadRel "confidential.seq" "confidential" "ExampleDB"
> confidential <- select "confidential" "ExampleDB"
```

These commands construct the relational representation of the information contained in the files `observed.seq` and `confidential.seq` and store them in `ExampleDB`, which is the data store for the session. We then select the relations from the data store to be used in our tests and computations. In the prototype tool, relations are represented as set-valued maps. Thus, the relations representing the observed information and the confidential information are given by:

```
> printRel observed
{
  "01" |-> ["12"]
  "02" |-> ["1"]
  "03" |-> ["16"]
  "04" |-> ["1"]
  "05" |-> ["17"]
  "06" |-> ["18"]
  "07" |-> ["11"]
```

```
  "08" |-> ["6"]
  "09" |-> ["17"]
  "10" |-> ["12"]
}
> printRel confidential
{
  "01" |-> ["3"]
  "02" |-> ["1"]
  "03" |-> ["4"]
  "04" |-> ["1"]
  "05" |-> ["5"]
  "06" |-> ["9"]
  "07" |-> ["2"]
  "08" |-> ["6"]
  "09" |-> ["5"]
  "10" |-> ["3"]
}
```

Once, the files have been loaded we can perform the first test: verifying the existence of an abstraction relation. The `test` function corresponds to Corollary 1.

```
> print (test confidential observed)
True
```

Here the result is `True`. This means that there exists an abstraction relation relating the confidential information to the information observed to be sent on the communication channel.

Now that we have verified that an abstraction relation does indeed exist, we can continue to compute the abstraction relation which relates the confidential information to the observed information. The `compute` function corresponds to Corollary 2. In the prototype tool, the universal relation $\mathbb{L}$ is represented as the *restricted universal relation on P and Q*, denoted $\mathbb{L}|_{P,Q}$, where P and Q are relations and

$$\mathbb{L}|_{P,Q} \stackrel{\text{def}}{=} \{(x, y) \mid x \in \text{ran}(P) \ \wedge \ y \in \text{ran}(Q)\}$$

This representation is required since, in the implementation of the prototype tool, we need to define the universal relation on a finite space. For the computation, the filter R from Corollary 2 is this universal relation restricted on the relations `confidential` and `observed` and is denoted by `top`. In our example, the relation `top` is given by:

```
> printRel top
{
  "1" |-> ["1","11","12","16","17","18","6"]
  "2" |-> ["1","11","12","16","17","18","6"]
  "3" |-> ["1","11","12","16","17","18","6"]
  "4" |-> ["1","11","12","16","17","18","6"]
  "5" |-> ["1","11","12","16","17","18","6"]
  "6" |-> ["1","11","12","16","17","18","6"]
  "9" |-> ["1","11","12","16","17","18","6"]
}
```

The computation of the abstraction relation is done with the `compute` function which corresponds to Corollary 2.

```
> printRel (compute confidential observed top)
{
  "1" |-> ["1"]
  "2" |-> ["11"]
  "3" |-> ["12"]
  "4" |-> ["16"]
  "5" |-> ["17"]
  "6" |-> ["6"]
  "9" |-> ["18"]
}
```

From this result, we can see that the abstraction relation is given by $X' = \{(1,1), (2,11), (3,12), (4,16), (5,17), (6,6), (9,18)\}$. In this case, we find that a 1 in the confidential information was transmitted as a 1 in the observed information, a 2 in the confidential information was transmitted as an 11 in the observed information, a 3 in the confidential information was transmitted as a 12 in the observed information, and so on.

With this small illustrative example, we are able to use the prototype tool to verify the existence of an abstraction relation relating the confidential information to the information observed to be sent on the communication channel and we are able to compute the abstraction relation.

Now, consider the case where agent A makes an error when representing the confidential information as outlined in Section 3.3.6. In this case, the monitor is configured with the same set of confidential information as in our running example, i.e., we still have `confidential.seq`. The difference is in what the monitor observes to be sent on the communication channel. Suppose that the information that the monitor records is the sequence $\langle 12, 1, 16, 12, 17, 18, 11, 6, 17, 12\rangle$. Then, this observed information is stored in a file which we call `observedError.seq`.

The observed information and confidential information are loaded in the prototype tool in the same way as described above so that the relations representing the observed information and the confidential information are given respectively by:

```
> printRel observedError
{
  "01" |-> ["12"]
  "02" |-> ["1"]
  "03" |-> ["16"]
  "04" |-> ["12"]
  "05" |-> ["17"]
  "06" |-> ["18"]
  "07" |-> ["11"]
  "08" |-> ["6"]
  "09" |-> ["17"]
  "10" |-> ["12"]
}
> printRel confidential
{
  "01" |-> ["3"]
  "02" |-> ["1"]
  "03" |-> ["4"]
  "04" |-> ["1"]
  "05" |-> ["5"]
  "06" |-> ["9"]
  "07" |-> ["2"]
```

```
  "08" |-> ["6"]
  "09" |-> ["5"]
  "10" |-> ["3"]
}
```

We automatically verify the existence of an abstraction relation using our prototype tool to apply Corollary 1 as we have done earlier to obtain

```
> print (test confidential observedError)
False
```

Here the result is `False`. Therefore, we can conclude that an abstraction relation relating the confidential information to the information observed to be sent on the communication channel does not exist.

Knowing the confidential information and the information that has been sent, we are able to automate the proposed technique to carry out a post-mortem analysis to detect whether the confidential information has been leaked in some form via covert channels. This sort of investigation relates to computer forensics investigations whereby, with some systematic assumptions, we are able to detect whether or not the information was leaked in one way or another. It is important to note that we do not necessarily care how the information was leaked, but simply whether it has been leaked.

5 Application in Cryptanalysis

Consider a scenario where an encrypted communication between two suspected criminals is intercepted. Suppose that an analyst is attempting to decipher the encrypted message and that the analyst has an intuition that the message contains some important pieces of information, i.e., a date, a location, a name, etc. Equipped with the proposed technique, the analyst can perform an investigation into the observed (intercepted) information transmitted between the suspected criminals. For instance, the analyst can run the test for an abstraction relation (Corollary 1) between the observed information and the information for which he/she suspects may have been sent. For example, the analyst may suspect that the transmission contains the suspected location of where a crime has taken place. If we allow these assumptions to form a confidential information, then the proposed technique can be used to search for an abstraction relation relating the suspected plain text to the intercepted cipher text message. If an abstraction relation can be found relating some plain text to the intercepted cipher text message, the analyst can begin to develop the cipher key that may have been used. This procedure is very much like the procedures developed by Alan Turing at Bletchley Park during World War II when deciphering the codes of the Enigma machine, which use cribs to analyse cipher texts [43].

Consider the scenario where an analyst is given a cipher text, N characters in length, and is asked to decrypt the message. We describe how the analyst can use the proposed technique for cryptanalysis on the given cipher. In performing the cryptanalysis, the analyst must begin by enumerating all of the cipher text characters (if necessary). We denote the set of all cipher text characters as $\mathbb{D}_C$ and the set of all plain text characters as $\mathbb{D}_P$. Assume that the analyst models the position of each character in the cipher text by $\mathbb{N}$. Then, the analyst is able to construct the relational representation of the cipher text as a relation $C \subseteq \mathbb{N} \times \mathbb{D}_C$. Next, the

analyst guesses a fragment of plain text which is suspected to occur in the cipher text. We denote the plain text fragment as $P = p_1 p_2 \ldots p_n$ where $\forall(i \mid 1 \leq i \leq n \leq N : p_i \in \mathbb{D}_P)$. The length of the plain text fragment is denoted by n. The idea is to check if there exists an abstraction relation between the plain text fragment and any cipher text fragment of length n for all positions in the cipher text. In order to test for an abstraction relation, the plain text fragment then needs to be represented as a set of relations $P_i \subseteq \mathbb{N} \times \mathbb{D}_P$ such that $\forall\big(i \mid 1 \leq i \leq N - n : P_i = \{(i, p_1), (i+1, p_2), \ldots, (i+n-1, p_n)\}\big)$. Similarly, we construct the corresponding fragments of the cipher text as relations $C_i \subseteq \mathbb{N} \times \mathbb{D}_Q$ such that $\forall\big(i \mid 1 \leq i \leq N - n : Q_i = \{(i, c_i), (i+1, c_{i+1}), \ldots, (i+n-1, c_{i+n-1})\}\big)$. Now, the analyst runs the test given in Corollary 1 for each pair of corresponding plain text, cipher text pairs, i.e., $P_i = C_i ; (C_i \backslash P_i) \quad \vee \quad C_i = P_i ; (P_i \backslash C_i)$ for $1 \leq i \leq N - n$. Each positive test result indicates that there exists an abstraction relation between the guessed plain text fragment and the corresponding cipher text fragment. This indicates the presence of a possible cipher key fragment. The analyst can compute each of the possible cipher key fragments by applying Corollary 2 for each (P_i, Q_i) pair yielding a positive test result. With the possible cipher key fragments, the analyst can apply the key fragment to the cipher text, which can reveal part of the plain text. With the additional information provided by applying the possible cipher key fragments, the analyst can generate a new, more refined guess at a plain text fragment suspected of occurring in the cipher text and repeat the process. The analyst may be able to infer a larger fragment of the message with some intuition of the neighbouring characters in the message. The process is very likely to converge on the complete cipher key thus decrypting the given cipher text message.

5.1 Case Study: Zodiac 408 Cipher

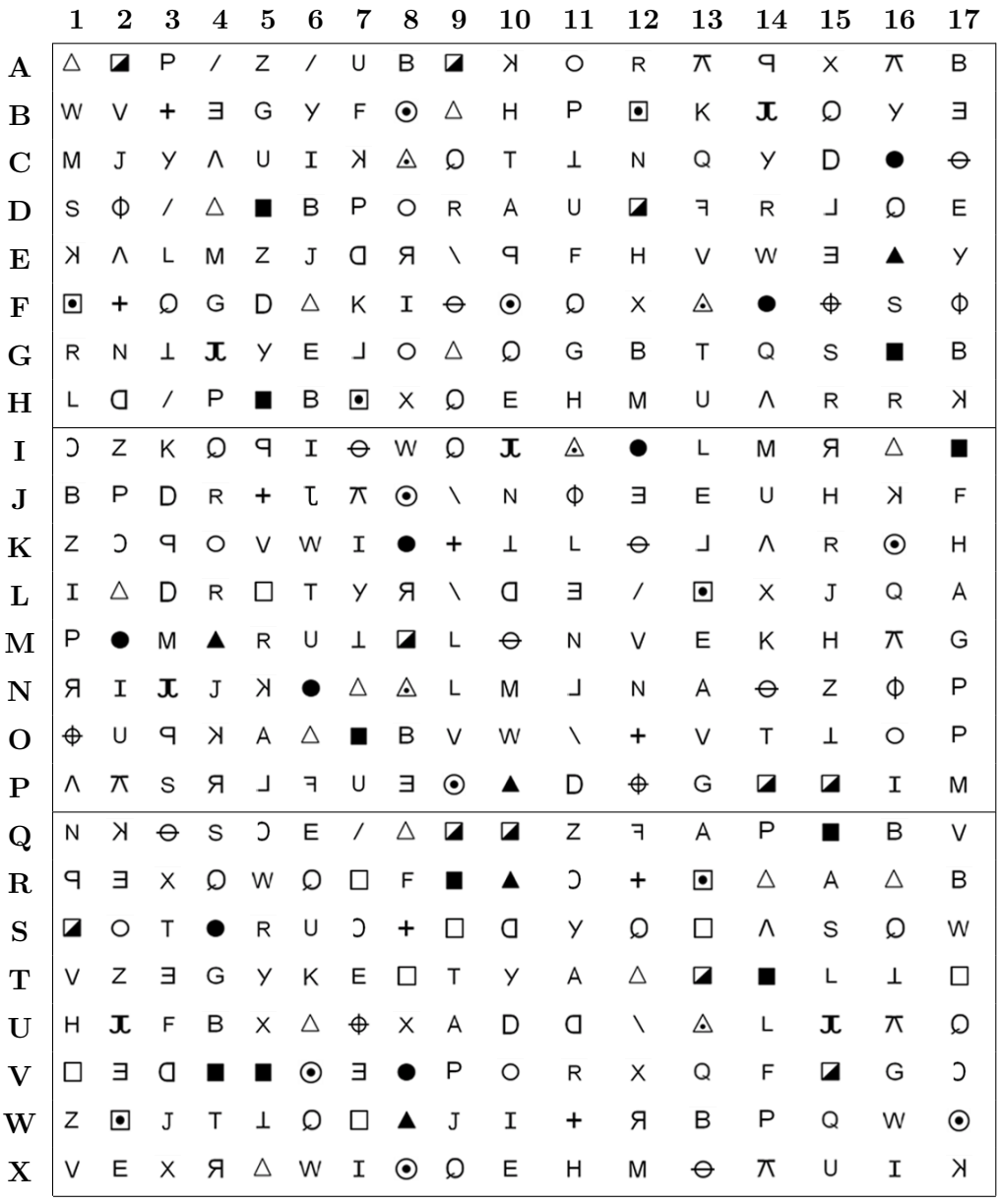
Using the prototype tool presented in Section 4, we have automated the process of applying the proposed technique to cryptanalysis. As an example, we study the Zodiac 408 cipher and show how the proposed technique, in conjunction with a known-plaintext attack, can break the cipher to uncover the message.

The Zodiac was a serial killer who terrorised Northern California in the late 1960's. The Zodiac sent four ciphers to local newspapers. The first cipher was separated into three different parts and each part was sent to three different newspapers: the Vallejo Times-Herald, the San Francisco Chronicle, and the San Francisco Examiner. The Zodiac requested each part be published on the front page of the respective newspapers such that the combination of all three parts formed a 408-character cipher, which was decrypted one week after it was received [5]. The Zodiac also sent a 340-character cipher that remains unsolved to this day. The case of the Zodiac killer remains open in Napa County, California [5].

In this paper, we examine only the 408-character cipher since it has a known solution and can be used for illustrative purposes. Table 1 shows the Zodiac 408 cipher in its entirety.

The Zodiac 408 cipher includes inaccuracies and errors which were made by the Zodiac himself when he transcribed the cipher symbols from the draft to the final version [51]. According to [51], in the Zodiac 408 cipher, **G6** should read F instead of E, **N13** should read H instead of N, and **U6** should read A instead of △. In our analysis, we have accounted for these errors and have made the necessary corrections.

Table 1: Zodiac 408 Cipher



We start by loading the prototype tool modules in the Glasgow Haskell Compiler's interactive environment (`ghci`) as follows:

```
> :load PrototypeTool
```

As preparation for the analysis, we first need to enumerate each of the 54 cipher characters so that we are able to represent them for use with the prototype tool. The enumeration is given in Table 2.

Table 2: Zodiac 408 Cipher Character Enumeration

1	△	10	R	19	W	28	⊡	37	M	46	D
2	◪	11	⊼	20	V	29	K	38	J	47	●
3	P	12	⅁	21	+	30	ℑ	39	Λ	48	⊖
4	/	13	X	22	∃	31	Ω	40	I	49	S
5	Z	14	Φ	23	G	32	■	41	◬	50	A
6	U	15	⅁	24	Y	33	⅃	42	T	51	E
7	B	16	L	25	F	34	⊏	43	⊥	52	Я
8	⋊	17	\	26	⊙	35	▲	44	N	53	⊕
9	O	18	Ↄ	27	H	36	ɾ	45	Q	54	□

We store the enumerated representation in a file for use with the prototype tool. From this point forward, we call the file containing the enumerated cipher text `cipher.seq`. We construct the relational representation of the information contained in the `cipher.seq` file and store it in the newly created data store, `Zodiac408`, by issuing the following commands:

```
> new "Zodiac408"
> loadRel "cipher.seq" "cipher" "Zodiac408"
> cipher <- select "cipher" "Zodiac408"
```

Now, the idea is to guess a known word or phrase that is likely to appear in the plaintext message corresponding to the Zodiac 408 cipher. Since, from the context, we know that cipher was written by a serial killer, it would be suspected that the author might have used words such as "kill", "killed", or "killing" or perhaps something like "zodiac" in some reference to himself. These would offer formidable starting points for the analysis. However, for simplicity and brevity, suppose that we have obtained a tip by some means (it is not important how) that suggests the plain text message contains the phrase "TO KILL SOMETHING GIVES ME THE MOST THRILL". In our representation of the phrase, we use only uppercase letters and ignore spaces as we are simply trying to find a readable plain text based on the assumption that spaces are not encoded. We generate a relation based on this phrase by issuing the following command with the prototype tool:

```
> let phrase = relFromString "TOKILLSOMETHINGGIVESMETHEMOSTTHRILL"
```

Then, the relation representing the plain text guess is given by:

```
> printRel phrase
{
  "01" |-> ["T"]
  "02" |-> ["O"]
  "03" |-> ["K"]
  "04" |-> ["I"]
  "05" |-> ["L"]
  "06" |-> ["L"]
  "07" |-> ["S"]
  "08" |-> ["O"]
  "09" |-> ["M"]
  "10" |-> ["E"]
  "11" |-> ["T"]
  "12" |-> ["H"]
  "13" |-> ["I"]
  "14" |-> ["N"]
  "15" |-> ["G"]
  "16" |-> ["G"]
  "17" |-> ["I"]
  "18" |-> ["V"]
  "19" |-> ["E"]
  "20" |-> ["S"]
  "21" |-> ["M"]
  "22" |-> ["E"]
  "23" |-> ["T"]
  "24" |-> ["H"]
  "25" |-> ["E"]
  "26" |-> ["M"]
  "27" |-> ["O"]
  "28" |-> ["S"]
  "29" |-> ["T"]
  "30" |-> ["T"]
  "31" |-> ["H"]
  "32" |-> ["R"]
  "33" |-> ["I"]
  "34" |-> ["L"]
  "35" |-> ["L"]
}
```

Now we are ready to apply the cryptanalysis technique described above. We use the following command:

```
> cryptanalysis phrase cipher
The 3 fragmented key possibilities are:

{
  "E" |-> ["19","24","25","39"]
  "G" |-> ["34","52"]
  "H" |-> ["1","35","37"]
  "I" |-> ["15","17","40","5"]
  "K" |-> ["2"]
  "L" |-> ["10","26","33","48"]
  "M" |-> ["20","28","8"]
```

```
  "N" |-> ["38"]
  "O" |-> ["21","51","6"]
  "R" |-> ["29"]
  "S" |-> ["27","31"]
  "T" |-> ["16","22","23","46","50"]
  "V" |-> ["12"]
}
{
  "E" |-> ["12","21","40","46"]
  "G" |-> ["22","35"]
  "H" |-> ["27","29","47"]
  "I" |-> ["20","24","37","49"]
  "K" |-> ["16"]
  "L" |-> ["10","14","38","5"]
  "M" |-> ["17","23","48"]
  "N" |-> ["19"]
  "O" |-> ["26","39","52"]
  "R" |-> ["53"]
  "S" |-> ["31","34"]
  "T" |-> ["1","13","25","41","8"]
  "V" |-> ["28"]
}
{
  "E" |-> ["12","19","5","51"]
  "G" |-> ["10"]
  "H" |-> ["37","48"]
  "I" |-> ["1","3","6","8"]
  "K" |-> ["4"]
  "L" |-> ["32","7"]
  "M" |-> ["31"]
  "N" |-> ["39"]
  "O" |-> ["13","30","34"]
  "R" |-> ["52"]
  "S" |-> ["28","29","41"]
  "T" |-> ["16","27","40","47"]
  "V" |-> ["18"]
}

The 3 possible plain texts are:
H K _ _ I _ O _ K M _ L _ V _ _ _
E M O T T E E L H S _ M R _ S E T
H N E E O I M _ S _ _ _ _ E T _ L
_ _ _ H _ _ _ _ L T O K I L L S O
M E T H I N G G I V E S M E T H E
M O S T T H R I L L S _ _ _ _ _ _
L _ _ _ E O L _ H S T _ _ _ _ _ _
T G _ _ _ _ M _ S O S H O E L L M
_ I R S V I L E S _ _ _ T H G H _
_ _ T L O _ _ L I _ _ T O O S M E
I _ V _ M E I _ O _ T L L E L L S
I H T L _ _ E G I G T _ M _ N _ T
_ _ H H L O _ K T L _ M O R S _ T
G I _ N M _ H H T H L S T L I _ _
```

```
_ O V M T H _ _ M E I O M _ _ _ _
E _ _ G L I O T L H T _ T K K I H
_ M L _ _ O _ H K K I I T _ _ _ M
V T _ S E S _ E _ H _ O M H T H _
K _ _ _ L O _ O _ G E S _ E _ S E
M I T T E R O _ _ E T H K _ T _ _
S _ E _ _ T _ _ T T G I _ T _ _ S
_ T G _ _ L T _ _ _ L _ _ E K T _
I M N _ _ S _ H N I O G _ _ _ E L
M O _ G H E I L S O S H L _ O I M

T _ _ _ L _ _ _ _ T _ L _ E T _ _
N I E G M I T O T H _ V H _ S I G
I L I O _ E T T S _ _ _ _ I E H M
I L _ T _ _ _ _ L _ _ _ _ L _ S _
T O K I L L S O M E T H I N G G I
V E S M E T H E M O S T T H R I L
L _ _ _ I _ _ _ T S M _ _ _ I _ _
K S _ _ _ _ V T S _ H I _ O L L T
_ L H S E E M N S _ T H K I O T _
_ _ E L E _ _ O M _ L G _ _ H T T
L _ E _ I N E H E _ K M _ O L O H
E T E L _ _ I O M S G _ V T L _ _
_ H I G L _ _ _ K M _ I _ H H _ M
O E _ L T H T G K I _ H _ M L L _
R _ E T _ T _ _ I N M E I _ _ _ _
O _ I O _ _ _ G O G E R M _ _ E I
_ T M I _ _ _ T _ _ L _ _ _ _ _ I
E G T S N S _ T _ G _ E V T _ T _
_ _ _ H L _ _ E _ S I S _ O I S N
I L G M I H _ _ _ I _ T _ _ K _ _
H _ T _ T _ R T _ E S M T K _ _ S
_ G S _ _ O G H _ _ L T _ T _ M _
L V L _ _ S _ G L E E O _ _ _ N O
I _ T O T N E O S _ H I M _ _ E T

I _ I K E K I L _ I _ G _ E O _ L
E _ _ _ _ _ _ _ I T I S S O M _ _
H _ _ N I T I S M _ _ _ _ _ _ T H
_ _ K I L L I _ G _ I _ _ G _ M E
I N T H E _ O R _ E _ T _ E _ _ _
S _ M _ _ I S T H _ M O S T _ _ _
G _ _ O _ E _ _ I M _ L _ _ _ L L
T O K I L L S O M E T H I N G G I
V E S M E T H E M O S T T H R I L
L I _ G _ _ _ _ _ _ _ _ E I T I _
E V E _ _ E T T _ _ T H _ N G _ T
T I _ G _ _ _ R _ O _ K S O _ _ _
I T H _ G I _ _ T H _ _ E S T _ _
R T O _ I T I _ T H _ T _ H E _ I
_ I E I _ I L L _ E _ _ _ _ _ _ I
N _ _ R _ _ I _ _ _ _ _ _ _ _ T H
_ I H _ V E K I _ _ E _ _ I L L _
```

```
E _ O M E M _ _ L _ V _ S I _ I L
_ _ _ T G I V _ _ O _ M _ N _ M E
_ E _ _ _ S E _ _ _ _ I _ L T _ _
T O _ L O _ _ O _ _ O _ S T O _ M
_ _ O L L _ _ T I _ G O _ _ _ _ V
E S _ _ _ M _ _ _ T _ R L I _ E _
_ E O R I E T _ M E T H H _ I T I
```

As a result of applying the cryptanalysis technique on the Zodiac 408 cipher, we find that we have three fragmented possibilities for the cipher key. In this case, by examining the three possible plain texts (which are generated by the prototype tool), it is easy to see that there is only one plain text which appears to make any sense; namely the third one. So, we can refine our phrase and try to run the cryptanalysis again. Based on the third possible plain text, we can see that our phrase is preceded by "LL" which might suggest that the word "ALL" comes before our original phrase. We can also see that succeeding our phrase, we have "I_G" which suggests the suffix "-ING". As a refined phrase, we can try the cryptanalysis with the phrase "ALL TO KILL SOMETHING GIVES ME THE MOST THRILLING". The process of performing the cryptanalysis is given below.

```
> let phrase2 = relFromString "ALLTOKILLSOMETHINGGIVESMETHEMOSTTHRILLING"
> cryptanalysis phrase2 cipher
The 1 fragmented key possibility is:

{
  "A" |-> ["49"]
  "E" |-> ["12","19","5","51"]
  "G" |-> ["10"]
  "H" |-> ["37","48"]
  "I" |-> ["1","3","6","8"]
  "K" |-> ["4"]
  "L" |-> ["32","7"]
  "M" |-> ["31"]
  "N" |-> ["39","46"]
  "O" |-> ["13","30","34"]
  "R" |-> ["52"]
  "S" |-> ["28","29","41"]
  "T" |-> ["16","27","40","47"]
  "V" |-> ["18"]
}

The 1 possible plain text is:
I _ I K E K I L _ I _ G _ E O _ L
E _ _ _ _ _ _ _ I T I S S O M _ _
H _ _ N I T I S M _ _ _ _ _ N T H
A _ K I L L I _ G _ I _ _ G _ M E
I N T H E _ O R _ E _ T _ E _ _ _
S _ M _ N I S T H _ M O S T _ A _
G _ _ O _ E _ _ I M _ L _ _ A L L
T O K I L L S O M E T H I N G G I
V E S M E T H E M O S T T H R I L
L I N G _ _ _ _ _ _ _ _ E I T I _
E V E _ _ E T T _ _ T H _ N G _ T
T I N G _ _ _ R _ O _ K S O _ _ _
I T H _ G I _ _ T H _ _ E S T _ _
```

```
R T O _ I T I _ T H _ T _ H E _ I
_ I E I _ I L L _ E _ _ _ _ _ _ I
N _ A R _ _ I _ _ _ N _ _ _ _ T H
_ I H A V E K I _ _ E _ _ I L L _
E _ O M E M _ _ L _ V _ S I _ I L
_ _ _ T G I V _ _ O _ M _ N A M E
_ E _ _ _ S E _ _ _ _ I _ L T _ _
T O _ L O _ _ O _ N O _ S T O _ M
_ _ O L L _ _ T I _ G O _ _ _ _ V
E S _ _ _ M _ _ _ T _ R L I _ E _
_ E O R I E T _ M E T H H _ I T I
```

Now, we are left with only one fragmented key. One can easily fill in many of the blanks in the possible plain text to reconstruct the original message, which is given in Table 3.

After substituting the letters and reconstructing the cipher symbols from their enumeration, we find that we have uncovered the cipher key for the Zodiac 408 cipher. The cipher key is given in Table 4.

In this section, we have demonstrated the application and automation of the proposed technique in the context of cryptanalysis. Using the Zodiac 408 cipher as an illustrative example, we have shown that in a cryptanalytic investigation where we may be able to perform a known-plaintext attack, we are able to uncover the encrypted message using our proposed technique.

6 Survey of the Literature

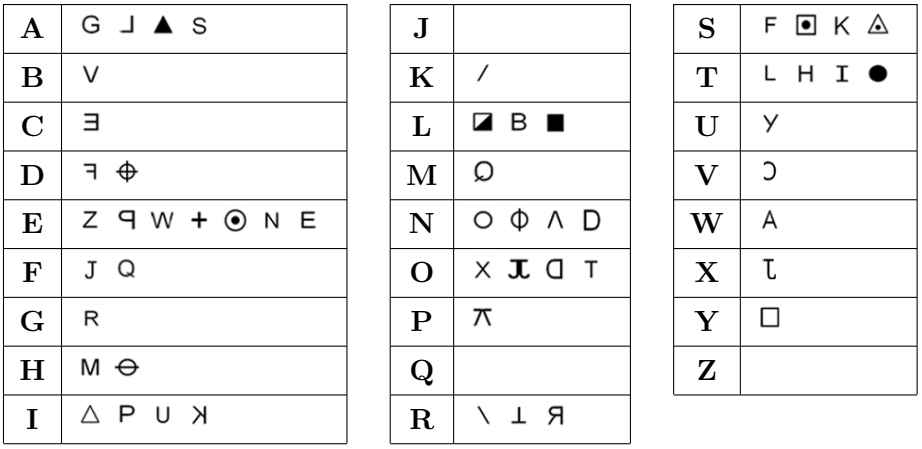
When it comes to eliminating the use of covert channels in computer systems, a variety of approaches have been proposed. Some approaches look at detecting the use of covert channels and some approaches look at preventing the use of covert channels, while there are very few approaches which aim to recover from the effects of covert channel use.

In [30], Nagatou and Watanabe present a technique for detecting the use of covert channels at run time. Security policies are enforced through flow control and access control mechanisms. The flow control mechanism compares the result of each system call into a system resource and the result of an emulator. If the results are different then it is considered that a covert channel occurred in the system and the monitor terminates the process that invoked the infracting system call. This technique is only able to enforce non-interference and non-inference policies. In the case of non-interference and non-inference, computer systems are modelled as machines with inputs and outputs, each classified as either low-level or high-level. A computer system has the non-interference property if and only if any sequence of low-level inputs will produce the same low outputs, regardless of what the high-level inputs are [8]. A computer system has the non-inference property if and only if an adversary cannot infer the value of a high-level output from low-level inputs [34]. The authors also admit that the monitor which they propose does not scale well since it would need to have emulators that have equal security levels and exploit many system resources. This technique also runs the risk of false positives, whereby the result of a system call into a system resource and the result of the emulator are different for reasons other than covert channels. One such example of this occurrence may be emulator failure. These noted weaknesses of this technique dramatically reduce the technique's ability to be used in many real world applications. However, the idea of monitoring the communication

Table 3: Zodiac 408 Plain Text

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
A	I	L	I	K	E	K	I	L	L	I	N	G	P	E	O	P	L
B	E	B	E	C	A	U	S	E	I	T	I	S	S	O	M	U	C
C	H	F	U	N	I	T	I	S	M	O	R	E	F	U	N	T	H
D	A	N	K	I	L	L	I	N	G	W	I	L	D	G	A	M	E
E	I	N	T	H	E	F	O	R	R	E	S	T	B	E	C	A	U
F	S	E	M	A	N	I	S	T	H	E	M	O	S	T	D	A	N
G	G	E	R	O	U	S	A	N	I	M	A	L	O	F	A	L	L
H	T	O	K	I	L	L	S	O	M	E	T	H	I	N	G	G	I
I	V	E	S	M	E	T	H	E	M	O	S	T	T	H	R	I	L
J	L	I	N	G	E	X	P	E	R	E	N	C	E	I	T	I	S
K	E	V	E	N	B	E	T	T	E	R	T	H	A	N	G	E	T
L	T	I	N	G	Y	O	U	R	R	O	C	K	S	O	F	F	W
M	I	T	H	A	G	I	R	L	T	H	E	B	E	S	T	P	A
N	R	T	O	F	I	T	I	S	T	H	A	T	W	H	E	N	I
O	D	I	E	I	W	I	L	L	B	E	R	E	B	O	R	N	I
P	N	P	A	R	A	D	I	C	E	A	N	D	A	L	L	T	H
Q	E	I	H	A	V	E	K	I	L	L	E	D	W	I	L	L	B
R	E	C	O	M	E	M	Y	S	L	A	V	E	S	I	W	I	L
S	L	N	O	T	G	I	V	E	Y	O	U	M	Y	N	A	M	E
T	B	E	C	A	U	S	E	Y	O	U	W	I	L	L	T	R	Y
U	T	O	S	L	O	W	D	O	W	N	O	R	S	T	O	P	M
V	Y	C	O	L	L	E	C	T	I	N	G	O	F	S	L	A	V
W	E	S	F	O	R	M	Y	A	F	T	E	R	L	I	F	E	E
X	B	E	O	R	I	E	T	E	M	E	T	H	H	P	I	T	I

Table 4: Zodiac 408 Cipher Key



among agents in the system is a good way to maintain a knowledge of the information flow of the system and has been adopted in Section 3.3.2, as it plays a large role in the development of the technique proposed in this paper.

In [23], Kemmerer describes a technique for detecting the use of covert channels in computer systems based on shared resources called the Shared Resource Matrix (SRM). The motivation for the SRM technique lies within the knowledge that the use of covert channels requires the collusion between an agent with the authorisation to signal or leak information to an unauthorised agent and that the authorisation is granted on system objects which may include file locks, device busy flags, the passing of time, etc. A matrix is constructed where the attributes of all shared resources are indicated in the row headings and the operation primitives, (i.e., `Write File`, `Read File`, `Lock File`, etc.), are indicated in the column headings. After all of the row and column headings are determined, one must determine, for each attribute (each row), whether the primitive indicated by the column heading modifies and/or references that attribute. This is done by carefully reviewing the description for each of the primitives, whether it is stated in natural language, formal specification, or implementation code. The generated matrix is then used to determine whether any covert channels exist. Kemmerer provides the following minimum criteria which must be satisfied in order to have a covert channel:

(i) The sending and receiving agents must have access to the same attribute of a shared resource.

(ii) There must be some means by which the sending agent can force the shared attribute to change.

(iii) There must be some means by which the receiving agent can detect the attribute change.

(iv) There must be some mechanism for initiating the communication between the sending and receiving agents and for sequencing the events correctly.

If each of these criteria are satisfied, then a covert channel exists. The advantages of the SRM technique include the ability to quickly discard attributes that do not meet the preliminary

criteria of being modified or referenced by an agent and the ability to provide a graphical design for developers in all stages of software design. However, the SRM technique is quite tedious and a little bit ad hoc in that the analyst must decipher scenarios in which the criteria might be satisfied.

Another technique for detecting covert channels in computer systems is Covert Flow Trees (CFTs). Presented by Kemmerer and Porras in [24] and [32], CFTs attempt to identify operation sequences that support either the direct or indirect ability of an agent to detect when an attribute has been modified. This means that CFTs aid in recognising when system attributes have been changed in some way by a sequence of operations. CFTs can be constructed automatically by providing the algorithm described in [24]. Once the CFT is constructed, the tree can be traversed to develop all possible operation sequences of the system. These operation sequences can then be analysed by developing hypothetical agents and system states that could use the operation sequences for covert communication. The analyst may assume that the sender and receiver share some mechanism whereby they can synchronise communication. CFTs are able to generate a comprehensive list of scenarios that could potentially support covert communication. The downfall of the CFT technique lies in the size of the CFTs that are generated and the scalability of the approach.

In [13, 14], Hélouët et al. propose a method for detecting potential covert channels using scenarios. The use of scenarios has several advantages. Scenarios are often the first information one can obtain about a system's behaviour since they are used to describe system requirements. Several recommendations [44, 46] ask to document the use of covert channels with such models. The idea is that from a scenario description of a system, a covert channel is modelled as a game where a pair of corrupted users, sender and receiver, try to send information while the rest of the protocol is attempting to prevent the information from being communicated. This scenario based approach only reveals "potential covert channels", the existence of which needs to be tested on a real implementation of the protocol.

According to [3], a system is separable (i.e., multilevel secure) if and only if it is behaviourally equivalent to a collection of single level systems that do not interact. In [3], Browne presents an approach called Mode Security. The idea is to organise the state transitions of a multilevel state machine into distinct sets called modes. The aim is to create a separable system. In essence, each machine mode is considered totally secure when considered in isolation of all other modes. This means that covert channels can only occur when the machine makes a transition from one mode to another. Therefore, by reducing the number of mode transitions in the system, one can reduce the number of potential covert channels in the system. Similarly, in [18], Jacob proposes a technique for detecting covert channels where the idea is to begin by making a list of all channels in a system. From this list, a new system is produced by "cutting" known channels from the system. This new system is checked for separability. If the new system is separable, then there are no covert channels, otherwise, at least one covert channel exists. The downfall of Jacob's technique is that it does not detect covert channels completely dependent on known channels. The major concern with approaches for mitigating covert channels based on separability is that these approaches are not universally applicable to all systems.

In [1], Andrews and Reitman provide an axiomatic definition for information flow in sequential programs, with particular emphasis on proof rules for programs containing assignment, alternation, iteration, composition, and procedure calls. The definition provided closely resembles Hoare's deductive system for functional correctness found in [15]. The axiomatic approach of Andrews and Reitman analyses programs looking for information flows which

violate the security policy of the system. A similar approach was taken by Sabri et al. in [38], where an amended version of Hoare logic was used to verify the satisfiability of security policies in communication protocols.

Since the confinement notion introduced by Lampson in [26], more and more approaches to detect illegal information flows have been proposed. A short while after Lampson, in [8], Goguen and Meseguer defined the existence of covert channels through non-interference properties. Numerous approaches to non-interference have been proposed. For example, in [47], Volpano and Smith describe the idea of non-interference through typing where a system contains interference if it cannot be correctly typed and in [28], Lowe describes non-interference using process algebra. The notion of non-interference is questioned in [37] since the transfer of a single bit of information causes a non-interference violation. According to [13], it is often the case that non-interference approaches attempt to classify data and processes of a system according to two security levels: high and low. However, it may not always be the case that there are only two security levels. This leads to a fundamental restriction of the use of non-interference properties to define the existence of covert channels in a system.

A wide variety of prevention schemes for the use of covert channels in computer systems have been proposed. One such approach is through information theoretic techniques such as channel capacity analysis. In [28], [29], and [41], mechanisms for computing the capacity of covert channels in computer systems are presented where the idea is that if the capacity of a covert channel can be reduced to a reasonably small rate, then the channel is rendered unusable as a means of effectively transferring information. The guidelines outlined in [44] and [46], state that covert channels with capacities of less than one bit per second are usually considered acceptable; while a capacity of more than 100 bits per second is considered unacceptable. One such method, developed by the United States Naval Research Laboratory, is called the Pump. Described in [22] and [27], the Pump lets information pass from a low level system to one at a higher level. The motivation comes from the idea that acknowledgements are required for reliable communication. If a higher level system passed acknowledgements directly to a lower level system, then the higher level system could pass high information by altering acknowledgement delays. In order to minimise such a covert channel, the Pump decouples the acknowledgement stream by inserting random delays. With consideration on overall performance of the system in mind, the Pump uses statistical averages to compute the delay time which it inserts into the communication stream. It is admitted in [27] that this method cannot handle a large state space which proves to be its major flaw. A number of additional prevention schemes take probabilistic approaches to covert channel mitigation. For example, in [11], Grusho et al. assume that for a secure transmission, covert channels will exploit a manipulation of the probability distribution parameters of the sent message sequence.

Although there are many existing techniques which aid in the fight against covert channels, there seems to be no single technique which can handle any type of covert channel in any type of system. Many of the existing techniques, based on mathematics in particular, seem to target specific types of covert channels. However, through the use of an abstract mathematical model, the proposed technique attempts to encompass covert channel communication in general.

7 Discussion

It has been a general assumption that it is impossible to completely eliminate covert channels from open systems. Any given open system typically contains several covert communication channels [9]. Many covert channels in computer systems arise from the use of shared resources. In order to completely eliminate covert channels, one would need to remove all contention for shared resources which leads to an inefficient utilisation of system resources and an unacceptable reduction in system performance. The detection and prevention of covert channels has been deemed challenging since the objects that are being used to hold the information being transferred are not normally seen as data objects i.e., buffer size, device flags, the passing of time, etc. We require a comprehensive and systematic way to model, detect, and prevent the use of covert channels without reducing the performance of the system to an unacceptable level.

In addressing covert channel threats, two challenges are distinguished: detection of covert channels and prevention of covert channels. In detecting covert channels, we ought to strive to develop techniques to identify covert channels in a systematic and comprehensive way. We must uncover the use of covert channels efficiently while minimising the number of false positives. We would like to provide some measure of assurance in the detection techniques being used. In covert channel prevention, we should determine ways to remove covert channels or at least find ways to restrict the use of covert channels without degrading the performance of the system to an unacceptable level. We ought to balance the tradeoff between system security and system performance which may not always be the most trivial of decisions.

The proposed relational model of covert channels offers simplicity when carrying out the computations required in the detection of confidential information leakage via covert channels. Since a stream is a discrete sequence of data, indexed by time, a stream provides major advantages in that it allows us to take intervals of data from the channel and examine each interval, leading to computations of finite relations rather than infinite ones. As well as gaining simplicity from the use of a stream representation, we also gain simplicity from the use of relations. Relations are simple mathematical concepts. They also offer a certain level of abstraction in the model of covert channels which gives much more power and flexibility in the ability to model particular types of covert channels.

In this paper, we are developing investigative support for confidentiality. This involves looking at covert channel communication from a digital forensics perspective. By its very nature, digital forensics is analysis after-the-fact [42]. Hence, the primary focus of a digital forensics investigation is placed on detection, that is, to prove that some form of violation of the security policy has taken place, despite that it seems the policy is being respected. Therefore, since we are dealing with analysis after-the-fact, performance is not a major consideration when developing detection mechanisms for covert channel communication. We simply need the analysis to be done in a reasonable amount of time.

The use of a covert channel detection technique in a computer forensics context is a new concept for generating investigative support for confidentiality. To the best of our knowledge, an application of a mathematical-based covert channel detection technique for computer forensics investigations is non-existent in the literature. The importance and necessity for this type of application seems to be growing day by day. Covert channels are a reality and are being used in real-world scenarios to smuggle information.

With the ongoing threat of insiders with malicious interest to cause harm or inappropriately access and divulge information, we must strive to devise new support to investigate

those individuals who may be responsible for the breach of confidentiality. As an example, we can examine the recent investigation in the United States of America with regards to a Russian spy ring. The spies were allegedly using various forms of covert communication including steganography, covert channels via e-mail protocols and even Morse Code-like radio signals [49]. This example stresses the real threat of covert communication on security. The implications of the use of covert channel communication on the scale of the international espionage highlights the importance of developing techniques to detect and prevent the use of covert channels in computer systems.

As the complexity of the covert channels increases and as the amount of information that is transmitted increases, will the proposed tests and computations be able to be computed in a reasonable amount of time? We examine the worst-case theoretical complexity of the tests and computations associated with the proposed technique. Let n be the cardinality of the relation. According to [31], the computational complexity of unary operators in relational algebra such as the complement and converse operators is $O(n)$. The computational complexity of binary operators in relational algebra is $O(n \log n)$ if each tuple of the first relation should be compared with each tuple in the second relation. This includes the set operators, union and intersection, as well as the comparison operations, equality and containment. According to [48], the computational complexity of relational composition is $O(n^3)$. By applying the rules for sequential composition and conditional execution [33], we arrive at the computational complexity for the tests and computations involved in the proposed technique. The computational complexity of the test described by Corollary 1 is $O(n^3)$. The complexity of the computations described by Corollary 2 and Corollary 3 are both $O(n^3)$.

The $O(n^3)$ complexity arises from the need to compute the composition of relations in each of the tests and computations. As the complexity of relational composition is the most expensive operation, it dominates the computational complexity of the algorithms for verifying the existence of an abstraction relation and for computing the abstraction relation if it does in fact exist. Further investigation into possible ways for improving the theoretical worst-case complexity for the tests and computations of the proposed technique is left as future work.

8 Conclusion and Future Work

In this paper, we presented a technique for detecting confidential information leakage via covert channels based on relational algebra. The technique does not rely on heuristics to uncover the use of covert channels. It gives a more formal and rigorous approach and offers a degree of simplicity. The technique provides tests to verify the existence of an abstraction relation and computations to find the abstraction relation if it exists. These tests and computations are expandable allowing for the technique to handle complex scenarios which may involve modulating the confidential information, for example. We also presented an application of the proposed technique in the area of cryptanalysis and discussed the use and development of tools to aid in the automation of the proposed technique and its applications.

The detection technique proposed in this paper does have some drawbacks. First and foremost, the tests can be averted under some conditions, particularly when there is an inconsistency between the confidential information and the information that is observed to be sent on the communication channel. Although this generally means that an abstraction relation does not exist which relates the confidential information to the observed information, it is possible that we can still find part of an abstraction relation which relates a large portion

of the confidential information to the observed information. It is also unknown how well the technique scales with larger systems. An empirical study is needed to address this point. In addition, the technique performs a post-mortem analysis of the communicated information. This can be seen as a weakness of the technique since the damage may already be done in terms of confidential information being leaked and falling into the wrong hands. It would perhaps be better if the analysis could be done in real-time.

Currently, it is unclear how communication channels can be effectively sampled for random testing to determine if any confidential information is being leaked. When we sample a large stream of information, it is possible that we will be sampling a portion of the communication stream which was leaking confidential information. However, rather than the sample containing the confidential information in its entirety, we may only have a portion of it. There is a need to be able to detect whether a part of the confidential information is contained in the sampled communication stream. Also, the proposed detection technique handles only a specific set of covert channels, i.e., protocol-based covert channels where the common knowledge is the sequence of the information (time). We would like to extend the proposed technique in order to tackle the most general covert channel possible where we have communication consisting of a combination of environmental and protocol-based knowledge.

More research into the applicability of the proposed detection technique for computer forensics investigations is needed. This research will likely involve more exploration into developing alternative tools and mechanisms for investigative support for confidentiality. The tests and computations related to the proposed detection technique are currently automated using our prototype tool. The use of the prototype tool to handle large scale covert channels needs to be examined further. However, it is possible that using Binary Decision Diagrams (BDDs) to represent relations, one can expect to handle fragments of sizes about 2^{32}; this size is tested with the BuDDy BDD library [20]. The continuing development of a more sophisticated and configurable automation system to handle large scale covert channels and to implement the capabilities of the communication monitors presented in Section 3.3.2 would be ideal.

References

[1] G. Andrews and R. Reitman. An axiomatic approach to information flow in programs. *ACM Transactions on Programming Languages and Systems*, 2(1):56 – 76, January 1980.

[2] M. Bishop. *Computer Security: Art and Science*. Addison Wesley, Boston, MA, November 2002.

[3] R. Browne. Mode security: An infrastructure for covert channel suppression. In *Proceedings of the 1994 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 39–55, Los Almitos, CA, USA, 1994.

[4] S. Cabuk, C. Brodley, and C. Shields. IP covert channel detection. *ACM Transactions on Information and Systems Security*, 12(4), April 2009.

[5] T. Dao. Analysis of the zodiac 340-cipher. Master's thesis, San Jose State University, December 2007.

[6] H. Furusawa and W. Kahl. A study on symmetric quotients. Technical Report 1998-06, Fakultät für Informatik, Universität der Bundeswehr München, December 1998.

[7] J. Giffin, R. Greenstadt, P. Litwack, and R. Tibbetts. Covert messaging through tcp timestamps. In *Proceedings of the Privacy Enhancing Technologies Workshop*, PET, pages 194–208, April 2002.

[8] J. Goguen and J. Meseguer. Security policies and security models. In *Proceedings of the 1982 Symposium on Security and Privacy*, pages 11–20, New York, NY, USA, 1982.

[9] J. W. Gray. Countermeasures and tradeoffs for a class of covert timing channels. Technical Report HKUST-CS94-18, Hong Kong University of Science and Technology, 2000.

[10] D. Gries and F. Schenider. *A Logical Approach to Discrete Math.* Springer Texts And Monographs In Computer Science. Springer-Verlag, New York, 1993.

[11] A. Grusho, A. Kniazev, and E. Timonina. Detection of illegal information flow. In V. Gorodetsky, I. Kotenko, and V. Skormin, editors, *Proceedings of the 3rd International Workshop on Mathematical Methods, Models, and Architectures for Computer Networked Security*, volume 3685 of *Lecture Notes in Computer Science*, pages 235–244. Springer Berlin / Heidelberg, Berlin, Germany, 2005.

[12] L. Hélouët and A. Roumy. Covert channel detection using information theory. In K. Chatzikokolakis and V. Cortie, editors, *Proceedings of 8th International Workshop on Security Issues in Concurrency*, SecCo 2010, pages 34–51, August 2010.

[13] L. Hélouët, M. Zeitoun, and A. Degorre. Scenarios and covert channels: Another game... *Electronic Notes in Theoretical Computer Science*, 119:93–116, 2005.

[14] L. Hélouët, M. Zeitoun, and C. Jard. Covert channels detection in protocols using scenarios. In *Proceedings of Security Protocols Verification*, SPV'03, pages 21–25, 2003.

[15] C. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, October 1969.

[16] C. Hoare and J. He. The weakest prespecification, Part I. *Fundamenta Informaticae*, 1986.

[17] C. Hoare and J. He. The weakest prespecification, Part II. *Fundamenta Informaticae*, 1986.

[18] J. Jacob. Separability and the detection of hidden channels. *Information Processing Letters*, 34(1):27–29, February 1990.

[19] R. Janicki and R. Khedri. On a formal semantics of tabular expressions. *Science of Computer Programming*, 39:189–213, March 2001.

[20] G. Janssen. A consumer report on BDD packages. In *Proceedings of the 16th Symposium on Integrated Circuits and Systems Design*, SBCCI 2003, pages 217 – 222, Washington, DC, USA, September 2003. IEEE Computer Society.

[21] J. Jaskolka and R. Khedri. Exploring covert channels. In *Proceedings of the 44th Hawaii International Conference on System Sciences*, HICSS-44, pages 1–10, Koloa, Kauai, HI, USA, January 2011.

[22] M. Kang and I. Moskowitz. A pump for rapid, reliable, secure communication. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 119–129, Fairfax, VA, USA, 1993.

[23] R. Kemmerer. Shared resource matrix methodology: An approach to identifying storage and timing channels. *ACM Transactions on Computer Systems*, 1(3):256–277, August 1983.

[24] R. Kemmerer and P. Porras. Covert flow trees: A visual approach to analyzing covert storage channels. *IEEE Transactions on Software Engineering*, 17(11):1166–1185, November 1991.

[25] R. Khedri. *Concurrence, Bisimulation et Équation d'Interface: Une Approche Relationnelle*. PhD thesis, Université Laval, April 1998.

[26] B. Lampson. A note on the confinement problem. *Communications of the ACM*, 16(10):613–615, October 1973.

[27] R. Lanotte, A. Maggiolo-Schettini, S. Tini, A. Troina, and E. Tronci. Automatic covert channel analysis of a multilevel secure component. In J. Lopez, S. Qing, and E. Okamoto, editors, *Proceedings of the 6th International Conference on Information and Communications Security*, volume 3269 of *Lecture Notes in Computer Science*, pages 249–261. Springer Berlin / Heidelberg, Berlin, Germany, 2004.

[28] G. Lowe. Quantifying information flow. In *Proceedings of the 15th IEEE Computer Security Foundations Workshop*, CSFW-15, pages 18–31, Los Alamitos, CA, USA, 2002. IEEE Computer Society.

[29] I. Moskowitz, S. Greenwald, and M. Kang. An analysis of the timed Z-channel. *IEEE Transactions on Information Theory*, 44(7):3162–3168, November 1998.

[30] N. Nagatou and T. Watanabe. Run-time detection of covert channels. In *Proceedings of the 1st International Conference on Availability, Reliability and Security*, ARES 2006, pages 577–584, Vienna, Austria, 2006.

[31] M. T. Özsu and P. Valduriez. *Principles of Distributed Database Systems*. Springer, third edition, 2011.

[32] P. Porras and R. Kemmerer. Covert flow trees: A technique for identifying and analyzing covert storage channels. In *Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 36–51, Los Alamitos, CA, USA, 1991. IEEE Computer Society.

[33] B. Priess. *Data Structures and Algorithms with Object-Oriented Design Patterns in Java*. Worldwide Series in Computer Science. Wiley, 1999.

[34] N. Ravi, M. Gruteser, and L. Iftode. Non-inference: An information flow control model for location-based services. In *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Systems*, pages 206–215, Piscataway, NJ, USA, 2006.

[35] R. Rowlingson. A ten step process for forensic readiness. *International Journal of Digital Evidence*, 2(3), Winter 2004.

[36] B. Russel. *Introduction to Mathematical Philosophy.* Routledge, 1993.

[37] P. Ryan, J. McLean, J. Millen, and V. Gligor. Non-interference: Who needs it? In *Proceedings of the 14th IEEE Workshop on Computer Security Foundation*, pages 237–238, Washington, DC, USA, 2001. IEEE Computer Society.

[38] K. Sabri, R. Khedri, and J. Jaskolka. Verification of information flow in agent-based systems. In G. Babin, P. Kropf, and M. Weiss, editors, *Proceedings of the 4th International MCETECH Conference on e-Technologies*, volume 26, pages 252–266. Lecture Notes in Business Information Processing, Springer Berlin / Heidelberg, May 2009.

[39] G. Schmidt and T. Ströhlein. *Relations and Graphs: Discrete Mathematics for Computer Science.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1993.

[40] C. Scott. Network covert channels: Review of current state and analysis of viability of the use of x.509 certificates for covert communications. Technical Report RHUL-MA-2008-11, Royal Holloway, University of London, January 2007.

[41] S. Shieh and A. Chen. Estimating and measuring covert channel bandwidth in multilevel secure operating systems. *Journal of Information Science and Engineering*, 15(1):91–106, 1999.

[42] S. Srinivasan. Security and privacy in the computer forensics context. In *Proceedings of the 2006 International Conference on Communication Technology*, pages 1–3, Piscataway, NJ, USA, November 2006. IEEE Computer Society.

[43] A. Turing. *The Essential Turing: Seminal Writings in Computing, Logic, Philosophy, Artificial Intelligence, and Artificial Life, Plus the Secrets of Enigma.* Oxford University Press, 2004.

[44] U.S.A. Department of Defense. *Trusted Computer System Evaluation Criteria (TCSEC).* Number DoD 5200.28-STD in Defense Department Rainbow Series (Orange Book). Department of Defense / National Computer Security Center, Fort George G. Meade, MD, USA, December 1985.

[45] U.S.A. Department of Homeland Security. A roadmap for cybersecurity research. Department of Homeland Security Science and Technology Directorate, Washington, DC, USA, November 2009.

[46] U.S.A. National Computer Security Center. *A Guide to Understanding Covert Channel Analysis of Trusted Systems.* Number NCSC-TG-030 in NSA/NCSC Rainbow Series (Light Pink Book). National Security Agency / National Computer Security Center, Fort George G. Meade, MD, USA, November 1993.

[47] D. Volpano and G. Smith. Eliminating covert flows with minimum typings. In *Proceedings of the 10th Computer Security Foundations Workshop*, pages 156–168, Los Alamitos, CA, USA, 1997.

[48] M. Wallace and S. Kollias. Two algorithms for fast incremental transitive closure of sparse fuzzy binary relations. *International Journal of Computational Methods*, 4(1):1–13, 2007.

[49] C. Williams. Russian spy ring bust uncovers tech toolkit. The Register, June 2010.

[50] S. Zander, G. Armitage, and P. Branch. Covert channels and countermeasures in computer network protocols. *IEEE Communications Magazine*, 45(12):136–142, December 2007.

[51] Zodiologists. Analysis of the zodiac killer's three part cipher (Z 408). Available: `http://www.zodiologists.com/z408_cipher_analysis.html` (Accessed: April 8, 2011), 2009.

A Proofs of Propositions and Corollaries

A.1 Detailed Proof of Proposition 5

(i) $P\backslash Q$ is surjective

$\iff$ ⟨ Formalisation ⟩

$\mathbb{L} = \mathbb{L}; (P\backslash Q)$

$\iff$ ⟨ Definition 11(ii) ⟩

$\mathbb{L} = \mathbb{L}; \overline{P^\smile; \overline{Q}}$

$\iff$ ⟨ P is a bijection $\iff$ $P^\smile$ is a mapping $\iff$ $P^\smile; \overline{S} = \overline{P^\smile; S}$ for all S ⟩

$\mathbb{L} = \mathbb{L}; \overline{P^\smile; Q}$

$\iff$ ⟨ Proposition 1(i) ⟩

$\mathbb{L} = \mathbb{L}; P^\smile; Q$

$\iff$ ⟨ P is total $\iff$ $P; \mathbb{L} = \mathbb{L}$ $\iff$ $\mathbb{L}; P^\smile = \mathbb{L}$ ⟩

$\mathbb{L} = \mathbb{L}; Q$

$\iff$ ⟨ Q is surjective $\iff$ $\mathbb{L}; Q = \mathbb{L}$ ⟩

$\mathbb{L} = \mathbb{L}$

$\iff$ ⟨ Identity of $=$ ⟩

true

(ii) $P\backslash Q = (Q\backslash P)^\smile$

$\iff$ ⟨ $(Q\backslash P)^\smile = P^\smile / Q^\smile$ ⟩

$P\backslash Q = P^\smile / Q^\smile$

$\iff$ ⟨ Definition 11(ii) & Definition 11(i) ⟩

$\overline{P^\smile; \overline{Q}} = \overline{\overline{P^\smile}; Q}$

$\iff$ ⟨ Complement both sides & Proposition 1(i) ⟩

$P^\smile; \overline{Q} = \overline{P^\smile}; Q$

$\iff$ ⟨ Antisymmetry ⟩

$P^\smile; \overline{Q} \subseteq \overline{P^\smile}; Q \ \wedge \ \overline{P^\smile}; Q \subseteq P^\smile; \overline{Q}$

$\iff$ ⟨ Proposition 3(i) ⟩

$P^\smile \subseteq (\overline{P^\smile}; Q)/\overline{Q} \ \wedge \ \overline{P^\smile} \subseteq (P^\smile; \overline{Q})/Q$

$\iff$ ⟨ Hypothesis: $P\backslash Q = (Q\backslash P)^\smile \iff P^\smile; \overline{Q} = \overline{P^\smile}; Q$ ⟩

$P^\smile \subseteq (\overline{P^\smile}; Q)/\overline{Q} \ \wedge \ \overline{P^\smile} \subseteq (\overline{P^\smile}; Q)/Q$

$\iff$ ⟨ Converse both sides & Proposition 1(ii) & Complement both sides & Proposition 1(i) ⟩
$P \subseteq \big[(\overline{P^\smile};Q)/\overline{Q}\big]^\smile \;\wedge\; \big[\overline{(\overline{P^\smile};Q)/Q}\big] \subseteq P^\smile$
$\iff$ ⟨ Definition 11(i) ⟩
$P \subseteq \big[\overline{\overline{P^\smile};Q};\overline{Q^\smile}\big]^\smile \;\wedge\; \big[\overline{\overline{\overline{P^\smile};Q};Q^\smile}\big] \subseteq P^\smile$
$\iff$ ⟨ Proposition 1(i) & Proposition 1(v) ⟩
$P \subseteq \overline{Q};\overline{Q^\smile;\overline{P}} \;\wedge\; \overline{\overline{P^\smile};Q};Q^\smile \subseteq P^\smile$
$\iff$ ⟨ Converse both sides & Proposition 1(ii) & Proposition 1(v) ⟩
$P \subseteq \overline{Q};\overline{Q^\smile;\overline{P}} \;\wedge\; Q;\overline{Q^\smile;\overline{P}} \subseteq P$
$\iff$ ⟨ $Q\backslash P$ is surjective $\iff$ $\overline{Q;\overline{Q^\smile;\overline{P}}} \subseteq \overline{Q};\overline{Q^\smile;\overline{P}}$ $\iff$ $\overline{\overline{Q};\overline{Q^\smile;\overline{P}}} \subseteq \overline{Q;\overline{Q^\smile;\overline{P}}}$ ⟩
$P \subseteq \overline{\overline{Q};\overline{Q^\smile;\overline{P}}} \subseteq \overline{\overline{Q;\overline{Q^\smile;\overline{P}}}} \;\wedge\; Q;\overline{Q^\smile;\overline{P}} \subseteq P$
$\Longrightarrow$ ⟨ Proposition 1(i) & Transitivity of $\subseteq$ ⟩
$P \subseteq Q;\overline{Q^\smile;\overline{P}} \;\wedge\; Q;\overline{Q^\smile;\overline{P}} \subseteq P$
$\iff$ ⟨ Definition 11(ii) ⟩
$P \subseteq Q;(Q\backslash P) \;\wedge\; Q;(Q\backslash P) \subseteq P$
$\iff$ ⟨ Proposition 4(iv) & Identity of $\wedge$ ⟩
$P \subseteq Q;(Q\backslash P)$

(iii) $P\backslash Q = (Q\backslash P)^\smile$
$\iff$ ⟨ Proposition 4(ii) ⟩
$P\backslash Q = P^\smile/Q^\smile$
$\iff$ ⟨ Definition 11(ii) & Definition 11(i) ⟩
$\overline{P^\smile;\overline{Q}} = \overline{\overline{P^\smile};Q}$
$\iff$ ⟨ Complement both sides & Proposition 1(i) ⟩
$P^\smile;\overline{Q} = \overline{P^\smile};Q$
$\iff$ ⟨ Antisymmetry ⟩
$P^\smile;\overline{Q} \subseteq \overline{P^\smile};Q \;\wedge\; \overline{P^\smile};Q \subseteq P^\smile;\overline{Q}$
$\iff$ ⟨ Proposition 3(ii) ⟩
$\overline{Q} \subseteq P^\smile\backslash(\overline{P^\smile};Q) \;\wedge\; Q \subseteq \overline{P^\smile}\backslash(P^\smile;\overline{Q})$
$\iff$ ⟨ Definition 11(ii) & Proposition 1(ii) ⟩
$\overline{Q} \subseteq \overline{P;\overline{\overline{P^\smile};Q}} \;\wedge\; Q \subseteq \overline{\overline{P};\overline{P^\smile;\overline{Q}}}$
$\iff$ ⟨ Complement both sides & Proposition 1(i) ⟩
$P;\overline{\overline{P^\smile};Q} \subseteq Q \;\wedge\; Q \subseteq \overline{\overline{P};\overline{P^\smile;\overline{Q}}}$
$\iff$ ⟨ Definition 11(ii) & Definition 11(i) ⟩
$P;(P^\smile/Q^\smile) \subseteq Q \;\wedge\; Q \subseteq \overline{\overline{P};(P\backslash Q)}$
$\iff$ ⟨ P is a bijection $\wedge$ Q is surjective $\Longrightarrow$ $P\backslash Q$ is surjective & $P\backslash Q$ is surjective $\iff$ $\overline{\overline{P};(P\backslash Q)} \subseteq \overline{\overline{P}};(P\backslash Q)$ ⟩

$$
\begin{array}{ll}
& P\,;(P^{\smile}/Q^{\smile}) \subseteq Q \ \wedge\ Q \subseteq \overline{\overline{P}\,;(P\backslash Q)} \subseteq \overline{\overline{P}}\,;(P\backslash Q) \\
\iff & \langle\ \text{Proposition 1(i)} \quad \& \quad \text{Transitivity of } \subseteq\ \rangle \\
& P\,;(P^{\smile}/Q^{\smile}) \subseteq Q \ \wedge\ Q \subseteq P\,;(P\backslash Q) \\
\iff & \langle\ \text{Hypothesis: } Q \subseteq P\,;(P\backslash Q)\ \rangle \\
& P\,;(P^{\smile}/Q^{\smile}) \subseteq Q \ \wedge\ \mathsf{true} \\
\iff & \langle\ \text{Identity of } \wedge \quad \& \quad \text{Proposition 3(i)}\ \rangle \\
& P \subseteq Q/(P^{\smile}/Q^{\smile}) \\
\iff & \langle\ \text{Definition 11(i)}\ \rangle \\
& P \subseteq \overline{\overline{Q}\,;(P^{\smile}/Q^{\smile})^{\smile}} \\
\iff & \langle\ \text{Proposition 4(i)}\ \rangle \\
& P \subseteq \overline{\overline{Q}\,;(Q\backslash P)} \\
\iff & \langle\ Q \text{ is a bijection} \quad \wedge \quad P \text{ is surjective} \quad \Longrightarrow \quad Q\backslash P \text{ is surjective} \ \& \\
& \qquad Q\backslash P \text{ is surjective} \iff \overline{\overline{Q}\,;(Q\backslash P)} \subseteq \overline{\overline{Q}}\,;(Q\backslash P)\ \rangle \\
& P \subseteq \overline{\overline{Q}\,;(Q\backslash P)} \subseteq \overline{\overline{Q}}\,;(Q\backslash P) \\
\iff & \langle\ \text{Proposition 1(i)} \quad \& \quad \text{Transitivity of } \subseteq\ \rangle \\
& P \subseteq Q\,;(Q\backslash P) \\
\iff & \langle\ \text{Hypothesis: } P \subseteq Q\,;(Q\backslash P)\ \rangle \\
& \mathsf{true}
\end{array}
$$

A.2 Detailed Proof of Proposition 6

$(\Longleftarrow)$ $Q = (Q/P)\,;P \implies X\,;P = Q$ has a solution

$$
\begin{array}{ll}
& X\,;P = Q \text{ has a solution} \\
\iff & \langle\ \text{Formalisation}\ \rangle \\
& \exists(X \ |:\ X\,;P = Q\,) \\
\iff & \langle\ \text{Antisymmetry}\ \rangle \\
& \exists(X \ |:\ X\,;P \subseteq Q \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{Proposition 3(i)}\ \rangle \\
& \exists(X \ |:\ X \subseteq Q/P \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{Definition of } \subseteq\ \rangle \\
& \exists(X \ |:\ \big(X = Q/P \ \vee\ X \subset Q/P\big) \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{Distributivity of } \exists \text{ over } \vee\ \rangle \\
& \exists(X \ |:\ X = Q/P \ \wedge\ Q \subseteq X\,;P\,) \ \vee\ \exists(X \ |:\ X \subset Q/P \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{Trading}\ \rangle \\
& \exists(X \mid X = Q/P \ :\ Q \subseteq X\,;P\,) \ \vee\ \exists(X \ |:\ X \subset Q/P \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{One-Point Axiom}\ \rangle \\
& Q \subseteq X\,;P\,[X \ :=\ Q/P] \ \vee\ \exists(X \ |:\ X \subset Q/P \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{Substitution}\ \rangle
\end{array}
$$

$$
\begin{array}{ll}
& Q \subseteq (Q/P)\,;P \ \vee\ \exists(X \ |:\ X \subset Q/P \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{Hypothesis: } Q = (Q/P)\,;P\ \rangle \\
& \mathsf{true} \ \vee\ \exists(X \ |:\ X \subset Q/P \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{Zero of } \vee\ \rangle \\
& \mathsf{true}
\end{array}
$$

$(\implies)$ $X\,;P = Q$ has a solution $\implies$ $Q = (Q/P)\,;P$

$$
\begin{array}{ll}
& X\,;P = Q \text{ has a solution} \\
\iff & \langle\ \text{Formalisation}\ \rangle \\
& \exists(X \ |:\ X\,;P = Q\,) \\
\iff & \langle\ \text{Antisymmetry}\ \rangle \\
& \exists(X \ |:\ X\,;P \subseteq Q \ \wedge\ Q \subseteq X\,;P\,) \\
\iff & \langle\ \text{Proposition 3(i)}\ \rangle \\
& \exists(X \ |:\ X \subseteq Q/P \ \wedge\ Q \subseteq X\,;P\,) \\
\implies & \langle\ \text{Isotony of } ;\ \rangle \\
& \exists(X \ |:\ X\,;P \subseteq (Q/P)\,;P \ \wedge\ Q \subseteq X\,;P\,) \\
\implies & \langle\ \text{Transitivity of } \subseteq \quad \& \quad \text{Idempotency of } \wedge\ \rangle \\
& \exists(X \ |:\ Q \subseteq X\,;P \subseteq (Q/P)\,;P \ \wedge\ Q \subseteq (Q/P)\,;P\,) \\
\iff & \langle\ \text{Distributivity of } \wedge \text{ over } \exists\ \rangle \\
& Q \subseteq (Q/P)\,;P \ \wedge\ \exists(X \ |:\ Q \subseteq X\,;P \subseteq (Q/P)\,;P\,) \\
\implies & \langle\ \text{Weakening}\ \rangle \\
& Q \subseteq (Q/P)\,;P \\
\implies & \langle\ \text{Proposition 4(iii)}\ \rangle \\
& Q \subseteq (Q/P)\,;P \subseteq Q \\
\iff & \langle\ \text{Antisymmetry}\ \rangle \\
& Q = (Q/P)\,;P
\end{array}
$$

A.3 Detailed Proof of Corollary 1

According to the problem formulation illustrated by Figure 5, we need to find solutions to either Equation 1 or Equation 2. Therefore,

$$
\begin{array}{ll}
& X\,;Q^{\smile} = P^{\smile} \text{ or } X^{\smile}\,;P^{\smile} = Q^{\smile} \text{ have solutions} \\
\iff & \langle\ \text{Proposition 6}\ \rangle \\
& P^{\smile} = (P^{\smile}/Q^{\smile})\,;Q^{\smile} \ \vee\ Q^{\smile} = (Q^{\smile}/P^{\smile})\,;P^{\smile} \\
\iff & \langle\ \text{Converse both sides \& Proposition 1(ii) \& Proposition 1(v) \& Proposition 4(i)}\ \rangle \\
& P = Q\,;(Q\backslash P) \ \vee\ Q = P\,;(P\backslash Q)
\end{array}
$$

A.4 Detailed Proof of Proposition 7

$(\Longleftarrow)$ $Q \subseteq \big(R \cap (Q/P)\big) ; P \implies X ; P = Q$ has $R \cap (Q/P)$ as a solution

$$
\begin{array}{ll}
& X ; P = Q \text{ has } R \cap (Q/P) \text{ as a solution} \\
\Longleftrightarrow & \langle\ X = R \cap (Q/P) \quad \& \quad \text{Substitution}\ \rangle \\
& \big(R \cap (Q/P)\big) ; P = Q \\
\Longleftrightarrow & \langle\ \text{Antisymmetry}\ \rangle \\
& \big(R \cap (Q/P)\big) ; P \subseteq Q \ \wedge\ Q \subseteq \big(R \cap (Q/P)\big) ; P \\
\Longleftrightarrow & \langle\ \text{Hypothesis: } Q \subseteq \big(R \cap (Q/P)\big) ; P\ \rangle \\
& \big(R \cap (Q/P)\big) ; P \subseteq Q \ \wedge\ \mathsf{true} \\
\Longleftrightarrow & \langle\ \text{Definition 11(i)} \quad \& \quad \text{Identity of } \wedge\ \rangle \\
& (R \cap \overline{\overline{Q} ; P^{\smile}}) ; P \subseteq Q \\
\Longleftrightarrow & \langle\ \text{Identity of } \wedge \quad \& \quad \text{Proposition 2} \quad \& \quad \text{Proposition 1(iv)}\ \rangle \\
& \overline{Q} ; P^{\smile} \subseteq (\overline{R} \cup \overline{Q} ; P^{\smile}) \\
\Longleftarrow & \langle\ \text{Weakening}\ \rangle \\
& \overline{Q} ; P^{\smile} \subseteq \overline{Q} ; P^{\smile} \\
\Longleftarrow & \langle\ \text{Reflexivity of } \subseteq\ \rangle \\
& \mathsf{true}
\end{array}
$$

$(\Longrightarrow)$ $X ; P = Q$ has $R \cap (Q/P)$ as a solution $\implies Q \subseteq \big(R \cap (Q/P)\big) ; P$

$$
\begin{array}{ll}
& X ; P = Q \ \wedge\ X = R \cap (Q/P) \\
\Longrightarrow & \langle\ \text{Substitution of } X\ \rangle \\
& \big(R \cap (Q/P)\big) ; P = Q \\
\Longleftrightarrow & \langle\ \text{Antisymmetry}\ \rangle \\
& \big(R \cap (Q/P)\big) ; P \subseteq Q \ \wedge\ Q \subseteq \big(R \cap (Q/P)\big) ; P \\
\Longrightarrow & \langle\ \text{Weakening}\ \rangle \\
& Q \subseteq \big(R \cap (Q/P)\big) ; P
\end{array}
$$

A.5 Detailed Proof of Corollary 2

(i)
$$
\begin{array}{ll}
& P \subseteq Q ; \big(R^{\smile} \cap (Q \backslash P)\big) \\
\Longleftrightarrow & \langle\ \text{Converse both sides \& Proposition 1(ii) \& Proposition 1(iv) \& Proposition 1(v)}\ \rangle \\
& P^{\smile} \subseteq \big(R \cap (Q \backslash P)^{\smile}\big) ; Q^{\smile} \\
\Longleftrightarrow & \langle\ \text{Proposition 7}\ \rangle \\
& \exists (X \mid X = R \cap (Q \backslash P)^{\smile} : X ; Q^{\smile} = P^{\smile})
\end{array}
$$

(ii) $Q \subseteq P\,;\big(R \cap (P\backslash Q)\big)$

$\Longleftrightarrow$ ⟨ Converse both sides & Proposition 1(ii) & Proposition 1(iv) & Proposition 1(v) ⟩

$Q^\smile \subseteq \big(R^\smile \cap (P\backslash Q)^\smile\big)\,;P^\smile$

$\Longleftrightarrow$ ⟨ Proposition 7 ⟩

$\exists(X \mid X = R \cap (P\backslash Q) : X^\smile\,;P^\smile = Q^\smile)$

(iii) $P \subseteq Q\,;\big(R^\smile \cap (Q\backslash P)\big) \wedge Q \subseteq P\,;\big(R \cap (P\backslash Q)\big)$

$\Longleftrightarrow$ ⟨ Converse both sides & Proposition 1(ii) & Proposition 1(iv) & Proposition 1(v) & Proposition 4(ii) ⟩

$P^\smile \subseteq \big(R \cap (P^\smile/Q^\smile)\big)\,;Q^\smile \wedge Q^\smile \subseteq \big(R^\smile \cap (Q^\smile/P^\smile)\big)\,;P^\smile$

$\Longleftrightarrow$ ⟨ Proposition 7 ⟩

$\exists(X \mid X = R \cap (P^\smile/Q^\smile) \wedge X^\smile = R^\smile \cap (Q^\smile/P^\smile) : X\,;Q^\smile = P^\smile \wedge X^\smile\,;P^\smile = Q^\smile)$

$\Longleftrightarrow$ ⟨ Converse both sides & Proposition 1(ii) ⟩

$\exists(X \mid X = R \cap (P^\smile/Q^\smile) \wedge X = R \cap (P\backslash Q) : X\,;Q^\smile = P^\smile \wedge X^\smile\,;P^\smile = Q^\smile)$

$\Longleftrightarrow$ ⟨ Golden Rule Axiom: $X = P \wedge X = Q \Longleftrightarrow X = (P \cup Q) \wedge X = (P \cap Q)$ ⟩

$\exists(X \mid X = \big[\big(R \cap (P^\smile/Q^\smile)\big) \cap \big(R \cap (P\backslash Q)\big)\big] \wedge X = \big[\big(R \cap (P^\smile/Q^\smile)\big) \cup \big(R \cap (P\backslash Q)\big)\big] : X\,;Q^\smile = P^\smile \wedge X^\smile\,;P^\smile = Q^\smile)$

$\Longleftrightarrow$ ⟨ Distributivity of $\cap$ of $\cup$ ⟩

$\exists(X \mid X = R \cap \big((P^\smile/Q^\smile) \cap (P\backslash Q)\big) \wedge X = R \cap \big((P^\smile/Q^\smile) \cup (P\backslash Q)\big) : X\,;Q^\smile = P^\smile \wedge X^\smile\,;P^\smile = Q^\smile)$

$\Longleftrightarrow$ ⟨ Golden Rule Axiom: $X = R \cap (P \cap Q) \wedge X = R \cap (P \cup Q) \Longleftrightarrow X = R \cap P \cap Q \wedge (P \cap Q) = (P \cup Q)$ ⟩

$\exists(X \mid X = R \cap \big((P^\smile/Q^\smile) \cap (P\backslash Q)\big) \wedge \big[\big((P\backslash Q) \cap (P^\smile/Q^\smile)\big) = \big((P\backslash Q) \cup (P^\smile/Q^\smile)\big)\big] : X\,;Q^\smile = P^\smile \wedge X^\smile\,;P^\smile = Q^\smile)$

$\Longleftrightarrow$ ⟨ Proposition 4(ii) ⟩

$\exists(X \mid X = R \cap \big((P^\smile/Q^\smile) \cap (P\backslash Q)\big) \wedge \mathsf{true} : X\,;Q^\smile = P^\smile \wedge X^\smile\,;P^\smile = Q^\smile)$

$\Longleftrightarrow$ ⟨ Definition 12 & Identity of $\wedge$ ⟩

$\exists(X \mid X = R \cap \mathrm{syq}(P,Q) : X\,;Q^\smile = P^\smile \wedge X^\smile\,;P^\smile = Q^\smile)$

A.6 Detailed Proof of Corollary 3

According to the problem formulation illustrated by Figure 5, we need to find solutions to Equation 1 and Equation 2.

$X = R \cap (P\backslash Q)$

$\Longleftrightarrow$ ⟨ Corollary 2 ⟩

$P \subseteq Q\,;\big(R^\smile \cap (Q\backslash P)\big) \wedge Q \subseteq P\,;\big(R \cap (P\backslash Q)\big)$

$\Longleftrightarrow$ ⟨ Converse both sides & Proposition 1(ii) & Proposition 1(iv) & Proposition 1(v) & Proposition 4(ii) ⟩

$P^\smile \subseteq \big(R \cap (P^\smile/Q^\smile)\big)\,;Q^\smile \wedge Q^\smile \subseteq \big(R^\smile \cap (Q^\smile/P^\smile)\big)\,;P^\smile$

$\iff$ $\langle$ Proposition 7 $\rangle$
$\exists(X \mid X = R \cap (P^\smile/Q^\smile) \wedge X^\smile = R^\smile \cap (Q^\smile/P^\smile) : X;Q^\smile = P^\smile \wedge X^\smile;P^\smile = Q^\smile)$
$\iff$ $\langle$ Converse both sides & Proposition 1(ii) $\rangle$
$\exists(X \mid X = R \cap (P^\smile/Q^\smile) \wedge X = R \cap (P\backslash Q) : X;Q^\smile = P^\smile \wedge X^\smile;P^\smile = Q^\smile)$
$\iff$ $\langle$ Proposition 4(ii) $\rangle$
$\exists(X \mid X = R \cap (Q\backslash P)^\smile \wedge X = R \cap (P\backslash Q) : X;Q^\smile = P^\smile \wedge X^\smile;P^\smile = Q^\smile)$
$\iff$ $\langle$ P is a bijection & Q is a bijection & $P \subseteq Q;(Q\backslash P)$ & $Q \subseteq P;(P\backslash Q)$ & Proposition 5(iii) $\rangle$
$\exists(X \mid X = R \cap (P\backslash Q) \wedge X = R \cap (P\backslash Q) : X;Q^\smile = P^\smile \wedge X^\smile;P^\smile = Q^\smile)$
$\iff$ $\langle$ Idempotency of $\wedge$ $\rangle$
$\exists(X \mid X = R \cap (P\backslash Q) : X;Q^\smile = P^\smile \wedge X^\smile;P^\smile = Q^\smile)$
$\iff$ $\langle$ Proposition 3(ii) $\rangle$
$\exists(X \mid X = R \cap (P\backslash Q) : X = P^\smile/Q^\smile \wedge X^\smile = Q^\smile/P^\smile)$
$\iff$ $\langle$ Converse both sides & Proposition 4(i) $\rangle$
$\exists(X \mid X = R \cap (P\backslash Q) : X = (Q\backslash P)^\smile \wedge X = P\backslash Q)$
$\iff$ $\langle$ Converse both sides & Proposition 4(i) $\rangle$
$\exists(X \mid X = R \cap (P\backslash Q) : (Q\backslash P)^\smile = P\backslash Q \wedge X = P\backslash Q)$
$\iff$ $\langle$ One-Point Axiom & $R = \mathbb{L}$ & Identity of $\cap$ $\rangle$
$\big((Q\backslash P)^\smile = P\backslash Q \wedge X = P\backslash Q\big)[X := P\backslash Q]$
$\iff$ $\langle$ Substitution $\rangle$
$(Q\backslash P)^\smile = P\backslash Q \wedge P\backslash Q = P\backslash Q$
$\iff$ $\langle$ Reflexivity of $=$ & Identity of $\wedge$ $\rangle$
$(Q\backslash P)^\smile = P\backslash Q$

A.7 Detailed Proof of Proposition 8

$\exists(X \mid\!: P;X = Q)$
$\iff$ $\langle$ Converse both sides & Proposition 1(v) $\rangle$
$\exists(X \mid\!: X^\smile;P^\smile = Q^\smile)$
$\iff$ $\langle$ Proposition 6 $\rangle$
$Q^\smile = (Q^\smile/P^\smile);P^\smile$
$\iff$ $\langle$ Converse both sides & Proposition 1(ii) & Proposition 1(v) & Proposition 4(i) $\rangle$
$Q = P;(P\backslash Q)$
$\iff$ $\langle$ Definition 11(ii) $\rangle$
$Q = P;\overline{P^\smile;\overline{Q}}$
$\implies$ $\langle$ M is total $\iff \mathbb{I} \subseteq M;M^\smile$ $\rangle$
$Q = P;\overline{P^\smile;\overline{Q}} \subseteq P;M;M^\smile;\overline{P^\smile;\overline{Q}}$
$\iff$ $\langle$ $M^\smile$ is deterministic $\iff$ $M^\smile$ is injective $\rangle$

$$
\begin{array}{ll}
 & Q = P\,;\overline{\overline{P^{\smile}}\,;\overline{Q}} \subseteq P\,;M\,;\overline{\overline{M^{\smile}}\,;\overline{P^{\smile}}\,;\overline{Q}} \\
\Longleftrightarrow & \langle\ \text{Proposition 1(v)}\ \rangle \\
 & Q = P\,;\overline{\overline{P^{\smile}}\,;\overline{Q}} \subseteq P\,;M\,;\overline{(P\,;M)^{\smile}\,;\overline{Q}} \\
\Longrightarrow & \langle\ \text{Transitivity of}\ \subseteq\ \rangle \\
 & Q \subseteq P\,;M\,;\overline{(P\,;M)^{\smile}\,;\overline{Q}} \\
\Longleftrightarrow & \langle\ \text{Definition 11(ii)} \quad \& \quad \text{Proposition 4(iii)}\ \rangle \\
 & Q \subseteq P\,;M\,;\big((P\,;M)\backslash Q\big) \subseteq Q \\
\Longleftrightarrow & \langle\ \text{Antisymmetry}\ \rangle \\
 & Q = P\,;M\,;\big((P\,;M)\backslash Q\big) \\
\Longleftrightarrow & \langle\ \text{Proposition 6}\ \rangle \\
 & \exists(Y\ \ |:\ P\,;M\,;Y = Q\,)
\end{array}
$$