Covert Channel Data Leakage Protection

A model for detecting and preventing data leakage through covert channels.

ADAM CORNELISSEN

UNIVERSITY OF TWENTE

TU/e Technische Universiteit Eindhoven University of Technology

Radboud University Nijmegen

Deloitte.

k3rckhoffs 1nst1tute

MASTER OF SCIENCE THESIS (647)

Supervisors:
prof. dr. B.P.F. Jacobs (Radboud University Nijmegen)
dhr. M. Knuiman (Deloitte Risk Services)
dhr. D. Wieringa (Deloitte Risk Services)

Colophon

Author:	Adam Cornelissen
Student number:	s0514268
Study program:	Computing Science
Graduation number:	647
Specialization:	Computer Security
Topic:	Covert Channel Data Leakage Prevention
University:	Radboud Universiteit Nijmegen (RU)
Faculty:	Faculteit Natuurwetenschappen, Wiskunde & Informatica (FNWI)
Institute:	Nijmeegs Instituut voor Informatica en Informatiekunde (NIII)
Address	Comeniuslaan 4 6525 HP Nijmegen Postbus 9102 6500 HC Nijmegen
Phone:	+31 (0)24-3616161
Fax:	+31 (0)24-3564606
Internet:	www.ru.nl
Place:	Nijmegen/Amstelveen (The Netherlands)
Version:	June 25, 2012
Supervisor Radboud University:	Bart Jacobs
1^{st} Supervisor Deloitte:	Martijn Knuiman
2^{nd} Supervisor Deloitte:	Derk Wieringa

Abstract

Sensitive, valuable, or secret enterprise information that leaks out of a company can make its way into the wrong hands, causing substantial loss for the involved businesses and undesirable situations for individuals. Data Leakage Prevention (DLP) refers to the practices of identifying, monitoring, and protecting sensitive information and is used to detect and prevent unauthorized use and transmission of this information. This research focused on DLP through covert channels; communication channels where the communication of sensitive information between two parties is done secretly via a channel that was not designed or intended for the sending of the information being sent. In our research we try to bridge the gap between theoretical information flow models, practical DLP solutions, and covert channel knowledge by creating a Covert Channel DLP model. In the process a machine learning Deep Content Analysis mode is developed, implementation concept database layout and implementation code described, and an existing product compared with our ideas.

Frequently used information flow models such as the Bell-LaPadula model, the Brewer and Nash model, and the Jericho approach do not incorporate the possibility of covert channels yet the concepts of 'need-to-know', conflicts of interest, and de-perimeterisation are useful in preventing unwanted transfer of information through these channels as well.

In order to be able to prevent or detect information leakage a DLP solution needs to have the functionality to detect sensitive information. Deep Content Analysis (DCA) techniques filter data according to a certain policy, which defines what (types of) content to protect. Traditional DCA techniques include rule-based/regular expressions, database fingerprinting, exact file matching, partial document matching, statistical analysis, conceptual/lexicon analyses, and category matching. New techniques such as searching in encrypted data, profiling, and machine learning are topics of research and experiments.

Modelling data leakage prevention consists of the communication channels themselves, how to handle transfer of information in certain situations (between insiders, outsiders, and shared resources), and the incorporation of a DLP system in these transfers. A covert channel DLP model requires additional information such as where covert channels can exist, how they can be detected, and how their existence can be prevented or limited as much as possible. Both the DLP model and the covert channel DLP model should allow for a situation where sensitive information can be transferred securely between authorized individuals and prevent unwanted situations from occurring. In comparison the covert channel model enforces the use of secured channels for all transfers and the employment of channel analysis techniques to stop transfers that go against the security policy.

The security policy contains information on what information is considered sensitive, how information and data transfers should be classified, and how incidents should be handled. Classification should be based on several factors including legal- and regulatory requirements, sensitivity and criticality, impact, and risks and threats.A data leakage incident response can vary in response time and appropriate action depending on the incident that occured.

The nature of a transfer impacts the effect content- and context characteristics have on a DLP system; if a file is transferred the information within that file is the main reference, while a data transfer via a covert channel depends much more on other characteristics such as sender- and receiver IP address, geographical location, and time of sending. The primary difficulty in an implementation of a covert channel DLP solution is the detection of the covert channel itself.

Comparing the methods, models, and techniques in this paper with a top of the line DLP product (according to Gartner) we notice that this (and similar) products do not consider covert channel data leakage an important factor; the focus area is aimed at employees who mistakenly or out of lack of other options cause data leakage events.

Detecting and preventing data leakage through covert channels is very difficult. Techniques for creating secure environments such as Deep Content Analysis, Channel Analysis, using industry level data- and channel encryption methods, and traditional protection mechanisms such as firewalls and IDS/IPS systems can help in securing information transfer and prevent/detect the existence of covert channels. Yet, the never ending resourcefulness of the human being will likely allow a determined, knowledgeable and sufficiently financed employee to leak information. The mantra here should be that the scope and level of protection should be specific and appropriate to the assets at risk.

Contents

List of Figures

List of Tables

Glossary

Abbreviation:	Explanation
CA	`Channel Analysis' Channel Analysis is the practice of inspecting the channel for oddities which might suggest that a covert channel exists. The difference with Deep Content Analysis is that not the information being sent is inspected but the channel itself.
CC	`Covert Channel' A communication channel is considered a covert channel when communication of sensitive information between two parties is done secretly via the channel that was not designed or intended for the sending of the information being sent which violates the security policy.
DCA	`Deep Content Analysis' Techniques to determine whether or not data contains sensitive information by inspecting the information itself. These techniques can also be used to classify documents.
DLP	`Date Leakage Prevention' DLP is a concept in which by encompassing different techniques the data flow inside and outside the corporation can be controlled with the goal to prevent sensitive information leakage.
DPA	`Differential Power Analysis' Technique to retrieve information about a secret key on a device (such as a smartcard) by investigating power differences at different stages of a cryptographic process.
IDS / IPS	`Intrusion Detection System' / `Intrusion Prevention System'. Technology related to the detection of (malicious) hacker attacks by observing traffic. Techniques range from the very basic signature correlation to the more advanced techniques such as deep packet inspection and behaviour analyses.
RFC	`Request for Comments' RFC documents describe the methods, behaviours, research, or innovations applicable to the working of the Internet and Internet-connected systems. The IETF adopts some of the proposals published as RFCs as Internet standards.

Table 1: Glossary

Chapter 1

Introduction

This chapter introduces the topic of the research and specifies the research questions.

Section 1.1 defines the topic. The goal is explicitly stated in section 1.2. The scope and strategy are elaborated upon in sections 1.3 and 1.4. The research questions are formulated in section 1.5. An overview of the whole thesis is given in section 1.6.

***

1.1 Topic.

Sensitive, valuable, or secret enterprise information that leaks out of a company can make its way into the wrong hands, causing substantial loss for the involved businesses and undesirable situations for individuals (e.g., identification theft). Besides the costs a company also faces reduced trust from customers, resulting in perhaps a bigger negative effect than the monetary loss alone. In many cases the company is breaking the law when data leakage has occurred due to inappropriate protection of (client) information such as health care records.

A recent incident in which the WikiLeaks organization (wikileaks.org) released 251,287 leaked diplomatic cables from 274 US embassies around the world has sparked the debate regarding data leakage protection. As a consequence of the cable leaks the US Army has forbidden the use of USB-sticks[1] and the Pentagon banned CD-burners[2]. This is just one example showing that both companies and governments are becoming more aware and concerned about possible data leakage.

`**Data Leakage Prevention**' (DLP) refers to the practices of identifying, monitoring, and protecting information. DLP is used to detect and prevent the unauthorized use and transmission of sensitive information.

Data Leakage Prevention is the research topic of this thesis, with a focus on the detection of `**Covert Channels**' (CC) and the prevention of information leaving the company through these channels.

See chapter 4 for more information on Covert Channels and chapter 5 for more information regarding Data Leakage Prevention.

[1] Security.nl: Leger VS verbiedt USB-sticks wegens WikiLeaks:
http://security.nl/artikel/35446/1/Leger_VS_verbiedt_USB-sticks_wegens_WikiLeaks.html

[2] Security.nl: Pentagon verbiedt cd-branders wegens WikiLeaks:
http://security.nl/artikel/35683/1/Pentagon_verbiedt_cd-branders_wegens_WikiLeaks.html

1.2 Goal.

A DLP solution is used to reduce the chance that private, secret, or sensitive information leaks out of an organization. Most DLP products available on the market today focus on accidental data leakage; unintentional actions that lead to data leakage. This research focuses on intentional data leakage; deliberate actions taken by persons within the organization with the intention to 'smuggle' sensitive information out of an organization.

The focus of this research is 'Covert Channels' and the prevention of information through these channels within the topic of DLP.

1.3 Scope.

The DLP field is very broad and thus it is necessary to define the scope of the research.

The study will look at technical side of DLP with a focus on covert channels. This includes looking at known covert channels, and how to detect these and other covert channels. To dive deeper into the prevention of data leakage theories on (secure) information flow, data protection methods, and content analysis techniques will be used.

Information Flow There are several information flow theories such as the Bell-LaPadula model, the first mathematical model of multilevel security policy used to define the concept of a secure state machine and modes of access and outlined rules of access (65) (a military-style model based on 'security clearances'), the Brewer and Nash model (also known as the Chinese Wall Model), a model which allows for dynamically changing access controls that protect against conflicts of interest (29), and the 'Jericho approach'. These models will be investigated to see how information flow is modelled conceptually and how such models can potentially be used to prevent data leakage and to detect covert channels.

Data Protection To protect data there are several options available such as encryption and obfuscation. Often attribute-based, role-based-, mandatory-, or discretionary access control systems are in place to restrict system access to authorized users only. Enterprise Digital Right Management (E-DRM), also referred to as Information Rights Management (IRM), technologies attempt to control use of digital media such as confidential files and documents. We look at these techniques to see to what extent they protect against data leakage.

Content Analysis Deep Content Analysis (DCA) techniques and content analysis concepts such as rule-based- / regular expressions, database fingerprinting, exact file matching, partial document matching, statistical analysis, conceptual/lexicon analyses, and category matching can be used to detect (parts of) information within data. These techniques are a key component within DLP solutions and are used to determine the 'sensitivity' of information and to detect the information within traffic. This can be done to either classify the information into categories (e.g., 'confidential', 'secret') or to detect sensitive information within (outgoing) data. When a sensitive piece of information is found leaving the company the DLP system triggers the appropriate alert and action to be taken.

The final goal is to create a link between the techniques, models, and theories of these different fields and construct a theory on covert channel data leakage detection and -prevention.

The concept 'DLP' can range from a relatively simple reconfigured IPS that uses keywords to block sensitive information, to whole enterprise-changing solutions that include classifying data, prioritizing security control objectives, organizational- and technical security policy creation, staff (awareness) training, documentation, accountability assignment, etcetera. In this thesis we focus on the more technical matters related to DLP and disregard practices such as awareness training and organizational policy creation which are essential when actually deploying DLP. Data leakage can occur at different levels (within the operating system, between applications, between computers, ..). Our focus will not be the communication within multi-user operating systems (e.g., information leakage between different security levels on a multilevel mainframe) but rather between different users over a network (e.g., data tunnelling over HTTP).

1.4 Strategy.

The first step is to accumulate and gain information from different theories, models, and methodologies on DLP, (un)secure communication, Deep Content Analysis, Information Flow models, and covert channel detection and -prevention to get an understanding of existing theories and techniques, as well as documentation on techniques such as steganography, encryption, obfuscation, data privacy, (industrial) espionage and related insiders which can give an insight in the methods that are used to secretly send information and on how to prevent it. The next step is combining the knowledge about DLP, Information Flow models, and covert channel detection and -prevention to create a theory on how to prevent and detect data leakage through covert channels. Once a model has been constructed we compare it against a real application.

1.5 Research Questions.

Main research question:

How can covert channel data leakage be detected and prevented?

Sub-questions related to defining concepts, detection and notation;

- What is information? (Chapter 2)
  When talking about 'information' it is important to specify what we mean by the term 'information'. Is information the same as 'raw data' or does it become information at a later stage?
  - When does 'data' become 'information'?
- How is 'sensitive' information separated from 'regular' information? (Chapter 2)
  Many models use some form of 'information clearance level' (or something similar). This separation of information sensitivity is an important part of a DLP process.
  - Besides 'content', must other factors be taken into account (e.g., 'context')?
  - How do Deep Content Analysis techniques detect information?
  - How do Deep Content Analysis techniques detect and separate 'sensitive' information from 'regular data'?
- What are information flow models? (Chapter 3)
  Information Flow models are represented by the combination of different stages connected by (directed) channels.
  - What information flow models can be used?
  - How do different models relate?
  - What could be their relation in respect to 'covert channels'?
- What are covert channels? (Chapter 4)
  Researching 'Data Leakage through Covert Channels' requires knowledge of the concept 'covert channels'.
  - What covert channels are known?
  - What theories/strategies/'best practices' are in place to detect covert channels?
  - How do covert channels fit into information flow models?

Sub-questions related to prevention;

- How can IT techniques contribute to privacy, classification, and loss prevention? (Chapter 2 and chapter 5)
  - Which data protection IT techniques (encryption, masking, ..) should be used to protect 'sensitive' information itself?
  - Which IT techniques can be used to restrict access to 'sensitive' information?

  - How can IT techniques be used to perform document classification?
  - How can IT techniques be used to prevent 'sensitive' information to leave the enterprise?
- What theories/strategies/'best practices' are in place to prevent covert channels? (Chapter 4)
- Which organizational techniques (policies, training) could be used, and when? (Chapter 5)
  Besides IT techniques it is important to create good policies and training to inform personal which and why certain procedures (e.g., e-mail scanning) are employed and to create a security-aware situation.

1.6 Thesis Layout.

Chapter 2 goes deeper into the meaning of 'information'. Questions such as 'What is information?' and 'How does information differ from data?' will be answered. How, if, and which information needs to be classified is also examined in this chapter. The chapter is relevant to the topic because it is necessary to know what should be inspected and protected in DLP systems.

In chapter 3 we will examine models that represent Information Flow, and the relation and differences between the different models. The topic of this chapter relates to DLP and CC in that trying to prevent data leakage includes knowing and limiting (possibly) unwanted information flows.

Chapter 4 describes covert channels. After defining the term 'covert channels' we continue to make the broad concept of covert channels more specific by dividing different types of covert channels into categories. The last part of this chapter is dedicated to known attacks (such as information leakage or OS detection) through covert channels.

Chapter 5 elaborates on the topic of Data Leakage Protection. Different types of data to protect, Content Analysis techniques, and DLP solutions are the contents of this chapter.

In chapter 6 a model is created to detect and prevent information leakage via covert channels.

The Security Policy and supporting processes define the practices for identifying, containing, responding and mitigating incidents. In chapter 7 we will focus on the DLP part of a Security policy. We will start on a high-level and look into the required governance structure, the classification and monitoring process, and the response handling. In a separate section we research the more implementation-focused 'security policy' as referenced in earlier chapters.

Chapter 8 compares the model created in chapter 6 with a real application ('Symantec Data Leakage Prevention 11') to see to what extent it employs the covert channel DLP model.

Related work on Data Leakage Prevention, Covert Channels, and related fields is stated in chapter 9 to give an overview of what has been done by other researchers.

Chapter 2

Information

In this chapter we will go deeper into the meaning of 'information'. What exactly are we trying to protect?

The key goal in this chapter is determining the difference between 'information' and 'data', and show some of the challenges when trying to prevent data leakage.

In the first section we will define what defines the term 'information' in comparison to 'data', 'knowledge' and 'wisdom'. Section 2.2 looks at several approaches to information protection and why they might not be sufficient on their own to prevent data leakage. Detection and classification ends this chapter.

***

2.1 Information Definition.

When we talk about 'data' and 'information' we talk about two different concepts. The two concepts are used quite often in the context of DLP, for example when determining what to protect; all data, or only some data?

In Ancient Greece 'information' was part of the concepts 'wisdom' and 'knowledge' where it took on the role of either 'know-how' or a more abstract perspective on the human condition (51):

"*Another human characteristic which the Greeks generally thought of as a virtue is wisdom (sophia) or knowledge (episteme). Two kinds or aspects of wisdom are particularly important for ethics. One is the practical ability to determine the correct course of action in a particular situation. This sort of wisdom is like the ability of a skilled craftsman to make the best use of material in creating an artistic product; it is a kind of 'know-how,' a skill. The second is more general, more abstract. It is a kind of perspective on the human condition, one which tells us who we are, what human life is like, and what our place is in the overall scheme of things. It is from this sort of wisdom that our understanding of the nature of the good life comes. Both kinds of wisdom contribute to the living of the good life: it helps to know both the end to aim at and the best means of reaching it.*"

The first Greek thinker to focus on wisdom as the virtue of central importance to ethics was Socrates; he identified virtue with knowledge, and treated knowledge as equivalent to wisdom although Prior notes that Socrates' view on the relationship between virtue and knowledge does not apply to knowledge in general but to knowledge of the end of human existence and the means to achieve it ('practical wisdom' (*phronesis*)). Although not all philosophers agreed with Socrates' identification of virtue with knowledge, they all agreed with the claim that wisdom or knowledge was an important virtue and a great aid to the living of 'the good life' (a life of excellence).

A more recent paper by Ackoff gives these two definitions for 'data' and 'information' in his paper 'From data to wisdom' (3):

Formulation 1. *Data is raw. It simply exists and has no significance beyond its existence (in and of itself). It can exist in any form, usable or not. It does not have meaning of itself.*

Formulation 2. *Information is data that has been given meaning by way of relational connection. This 'meaning' can be useful, but does not have to be.*

Besides these two concepts Ackoff states three more; Knowledge, Understanding, and Wisdom, and summarizes the relationship between these five abstractions as:

Type:	**Explanation:**
Data	Symbols.
Information	Data that are processed to be useful; answers "who/what/where/when".
Knowledge	Application of data and information; answers "how" questions.
Understanding	Appreciation of "why".
Wisdom	Evaluated understanding.

Table 2.1: Ackoff's differences from 'data' to 'wisdom'.

These definitions (at least 'knowledge' and 'wisdom') do not differ so much from the definitions given to these concepts in ancient Greece; 'knowledge' is more related to 'how' something is done (know-how), while 'wisdom' encompasses a deeper understanding and appreciation for knowledge.

Aamodt et al. note that the answer to questions such as 'What is information' has been a major problem of philosophers and scientists since ancient times (1). They state that it is beyond the scope of computer science to provide a general definition of these terms so they only define the term 'information' from a computational information processing system perspective. They limit themselves to a characterization of data, information, and knowledge from the perspective of development and operation of this type of systems, in particular decision support systems, including databases, information systems, and knowledge-based systems.

In the single agent decision-making process Aamodt et al. summarize the essential differences between the three concepts 'Data', 'Information', and 'Knowledge' as follows:

Formulation 3. *Data are syntactic entities: Data are patterns with no meaning; they are input to an interpretation process, i.e. to the initial step of decision making.*

Formulation 4. *Information is interpreted data: Information is data with meaning; it is the output from data interpretation as well as the input to, and output from, the knowledge-based process of decision making.*

Formulation 5. *Knowledge is learned information: Knowledge is information incorporated in an agent's reasoning resources, and made ready for active use within a decision process; it is the output of a learning process.*

Information within a company ranges from interpreted raw numbers on product production, quality reports, product blueprints, but also (regular) e-mail traffic and other documents. A difference between Ackoff and Aamodt is that Ackoff's definition specifically mentions that information does not need to be useful. While interpreted data might not be useful all the time it often is, or could be in the future. Information not used by one party might prove to be very useful for another.

In our paper we use the following definitions to define data and information based on the definitions given above:

Definition 1. Data is raw. It simply exists and has no significance beyond its existence (in and of itself). It can exist in any form, usable or not. It does not have meaning of itself.

Definition 2. Information is data that has been given meaning by interpretation of data.

Besides the difference between 'data' and 'information', we also use a difference definition for 'sensitive' information instead of 'regular' or 'non-sensitive' information.

We define sensitive information as;

Definition 3. Sensitive information is information or knowledge that, when leaked to unauthorized receivers, unacceptably increases the chance of damage being done to an involved party.

In this definition 'damage' can mean various things such as, but not limited to, financial loss, loss of brand value, loss of customers, privacy violations, or increased security risks. Whether or not the consequence of such damage is deemed 'unacceptable' has to be determined. This is in some cases handled by law (e.g., consumer privacy protection), the company that stores the sensitive information, or another third-party (e.g., organization 'watch dogs'). Involved parties include government agencies, companies, customers, and employees. To differentiate between 'regular' and 'sensitive' information analysis techniques such as 'Deep Content Analysis' can be applied (see section 5.6).

2.2 Information Protection

There are several commonly used techniques to protect information. The most common approaches can be divided in two categories; those that focus on the information itself and those that focus on the access to the information.

Encryption is a well-known example of direct information protection; information is encrypted using some encryption method and can only be decrypted when the correct credentials or necessary tokens are in place (e.g., user/password combination, or a smartcard).

Access control systems (either attribute-based, role-based-, mandatory,- or discretionary) or simple access control lists (ACL) can be used to determine which insiders (i.e., employees) have access to a certain piece of information, and what rights they have on that information. Here the protection is not directly connected to the information but rather on limiting the access to that information.

In this paper we focus on malicious insiders (people) who have legitimate access to the information. Access control systems and encryption alone do not always necessarily help to prevent information leakage; Once an insider (e.g., an employee) has gained access by having the appropriate access rights or by decrypting the information the 'protection phase' of this information is over. When no other mechanisms are in place, the obtained and decrypted information could easily be sent to outsiders. In a way, systems with no additional protection mechanisms rely on the integrity of employees who have legitimate access to the information. Figure 2.1 illustrates this.

Enterprise Digital Rights Management (E-DRM, sometimes also referred to as 'Information Rights Management') protects sensitive information by managing and enforcing access and usage rights to the information throughout its life cycle, no matter where the information is distributed. E-DRM systems often use encryption or isolation to prevent direct access to content, and enforce access- and usage rights by trusted DRM client software (which can be authenticated using a DRM license or policy servers with decryption keys and rights policies) (67). Some E-DRM solutions (such as *Display-Only File Server* (DOFS)) provide users a 'display image' of a file rather than the actual bits of the file, and many are equipped with screen capture/printer/copy&paste disable possibilities to provide a good measure against information theft. Weaknesses in E-DRM systems include the E-DRM client, and the fact that the information is sometimes stored on a computer that is completely under the attacker's control. Opponents of E-DRM solutions state that it is only a matter of time before the protection mechanism is broken and the rights enforcement checking is bypassed (53).

The idea of DLP solutions is to not to solely protect the information itself (which could be by means of the above mentioned protection methods) from unauthorized insiders directly, but to also prevent this information (in unprotected format) from leaving the company through authorized insiders (See chapter 5 on DLP). As such, DLP focuses on both the protection of the information (authorized obtainability of information) and the flow of information.

To regulate the flow of information between different parties to determine where information is (allowed to be) flowing several models have been created. This is discussed in chapter 3.

DLP solutions are not the only systems that are designed to protect the company. Several other common security systems are also designed to protect information or access to information, including firewalls, IDS/IPS systems, and proxies.

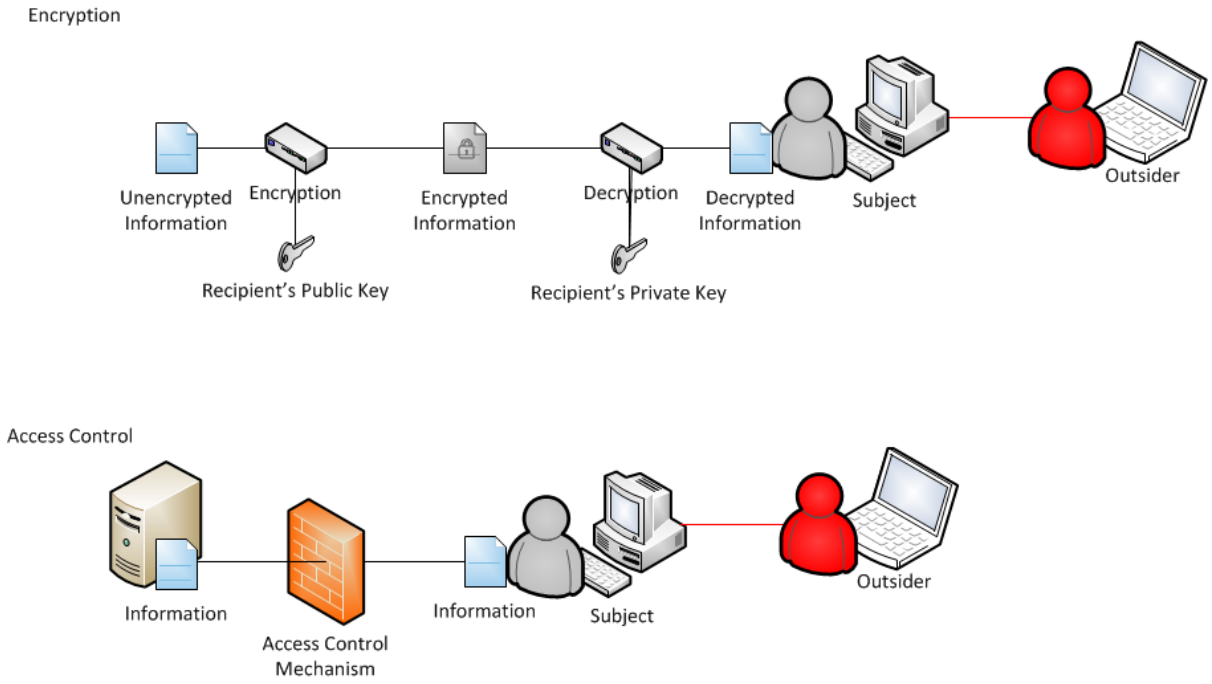


Figure 2.1: Encryption and Access Control failing to protect against data leakage when the sender is an insider with legitimate access to the information.

Firewalls

Firewalls are designed (mainly) to 'keep the bad guys out'. This is done by blocking unwanted packets and by preventing malicious or unknown users and applications to communicate to the network. In general there are two types of firewalls; firewalls which operate on the Network layer and firewalls which operate on the Application layer.

The Network layer firewall looks at the individual packets passing through and, using a set of firewall rules, decides whether or not it is allowed to pass. The Application layer firewall inspects packets from and to a certain application (such as a web server or an e-mail client). By analysing the data it decides whether or not to regard the traffic as 'unwanted'.

A mayor similarity between a Network layer firewall and DLP is that both use Packet Inspection. The difference is that a DLP solution doesn't only filter based on the context information (e.g., source IP, destination IP, port) but possibly also on the contents to check whether or not data is sensitive.

Application layer firewalls and DLP solutions both look at packets to and from applications, and can inspect these packets. The difference is that the firewall looks for potential harmful traffic. An example is a database which can be protected by an Application layer firewall to prevent SQL injection, database rootkits, and unauthorized information disclosure. A DLP solution does not look for potential harmful traffic but whether or not the outgoing data contains sensitive information.

Network layer firewalls can be further sub-divided into stateful and stateless groups, and Application layer firewalls can be divided into network-based Application firewalls and host-based Application firewalls. Either way, the mayor difference between DLP and any type of firewall is the intention; A firewall tries to keep the bad guys out of the network while a DLP solution tries to keep sensitive information within the network. When inspecting the data on the network this means that the firewall would check the incoming traffic and decide whether or not it is legal traffic, while a DLP solution inspects the outgoing traffic to check if the data does not contain sensitive information.

A DLP solution is not a firewall. Despite the differences there are techniques and practices that a DLP implementation can use that are often seen in firewalls:

- Layout of a security policy;

- Machine processable policy;
- Usage of whitelists / blacklists;
- Both DLP and Firewalls can be used for Network protection and Endpoint protection.

A DLP solution uses an implementation of the security policy to be able to decide which information can be sent to whom and over which channels. A firewall also uses an implementation of the security policy (a firewall policy) in which is stated how traffic should be handled, what should be blocked and what is allowed to pass through.

Some things that are clearly different;

- Focus on blocking intruder in a firewall versus blocking information leakage in a DLP solution;
- Mostly no packet content analysis in firewalls (apart from header information).

IDS/IPS

Intrusion Detection Systems (IDS) are systems designed to monitor events and identify possibly incidents (e.g., hack attacks). Besides monitoring and identification of threats IDS's are generally also able to reset TCP connections by issuing specially crafted packets after an attack begins. Some are also capable to communicate with firewalls to allow on-the-fly adjustment of firewall rule sets. An Intrusion Prevention System (IPS) differs from an IDS in the added ability to prevent malicious activities; In contrast to IDS sensors which watch network traffic as it passes by, the network traffic has to flow through an IPS system. This allows an IPS system to pull or drop traffic (17).

The most common technique used by IDS and IPS environments to identify incidents is signature comparison (sniffing the packets and comparing it with known attack patterns). Most DLP systems also use signature comparison (comparing data with known sensitive information). Another technique is based on statistics of the network traffic; alerts are raised whenever 'unusual' traffic is detected (e.g., traffic on rarely used ports, sudden increase of bandwidth, etcetera). Statistic analysis can also be used to detect covert channels.

A (sometimes contested) technique in use by IDS/IPS systems is Deep Packet Inspection (DPI) in which the data fields of a packet are also inspected to detect threats. DLP systems can use analysis techniques which can make use of Deep Packet Inspection.

Similarities between IDS/IPS and DLP solutions:

- Similar techniques to detect threats;
- Both can apply DPI;

Differences:

- Focus on preventing access to resources (or other functions) versus blocking information leakage;
- Blocking potentially harmful traffic versus blocking information leakage

Proxies

A proxy is typically used in organizations to check whether or not a client request for a resource (a file, web page, etc.) is allowed. When access is granted the proxy requests the resource and sends it to the client. Such systems are often applied to block access to websites that might be dangerous or not related to the business process (such as gambling websites), but also for speed reasons (proxy cache) and client anonymity. Often the only way to connect to the internet from within a company is through a proxy.

Proxies can also alter the traffic which can be useful for malware removal. Reverse SSL proxies can be used to sniff SSL traffic (if the necessary keys/certificates are in place).

Similarities between proxies and DLP solutions:

- Both manage access from insiders to (internal) resources;
- Both do not focus on preventing attacks from outsiders.

Differences:

- Focus on preventing access to resources (or other functions) versus blocking information leakage;
- A proxy is often the only way from a client to the internet (gateway);
- A proxy typically does not operate on the network layer (like firewalls) but on the application layer of the OSI model.

DLP

Information protection is the goal of DLP; at different stages (in use, in motion, and in rest, see also section 5.5) different protection- and detection techniques can be used in a DLP implementation. As such, DLP is a collection of existing information protection techniques, specifically focused on data leakage.

An example of this concept includes using encryption for sensitive information that is being transferred or stored, monitoring transfers using IDS/IPS systems, applying access control mechanisms (firewalls, proxies) for fetching information from shared resources, and E-DRM when processing sensitive information.

Another important part of DLP is the identification and classification of information; clearly, in order to protect sensitive information there is a need to know what information is (or should be) classified as such and where this information resides.

Data classification techniques can be used to determine who can have access to certain pieces of sensitive information. This classification can be done manually, but also (partially) automatic by means of inspecting the content and context (e.g., owner, creator) of the information. In the classification process data is inspected to see whether it contains information and if so the information is evaluated to determine whether or not it should be labelled `sensitive'.

Data classification techniques can also be used to detecting information flowing out of a company by comparing detected information with known sensitive information or assumed sensitive information. Because data classification is a very important stage in the information detection mechanism of a DLP system it is treated separately and in more detail in chapter 5.6.

Chapter 3

Information Flow Models

Data Leakage Prevention systems often use techniques to control information flow between (groups or types of) users. There are models for such flows of information. Some of these models are reflected upon within this chapter; the Bell-LaPadula model (section 3.1), the Chinese Wall model (section 3.2), and the Jericho approach (section 3.3).

Section 3.4 looks at the link between Information Flow models and our model.

3.1 Bell-LaPadula model.

The Bell-LaPadula model is the first mathematical model of multi-level security policy used to define the concept of a secure state machine and modes of access and outlined rules of access (65). Its goal is to identify the allowed ways of communication when confidentiality of information must be guaranteed. It is used in multi-level security solutions such as SELinux (besides Type Enforcement (TE) and Roles Based Access Control (RBAC)). A multi-level security system implements both hierarchical security and multilateral (categorical) separation. The Bell-LaPadula model uses security levels on objects (such as 'Unclassified', 'Confidential', 'Secret', and 'Top Secret', so-called 'Classifications') and security levels on insiders (called 'Clearance'), and categories.

To view a piece of information, the insider should have the 'need-to-know' for all categories of the information as well as the appropriate security clearance; the clearance of the insider should be at least equal to or higher than the classification of the information, a process called 'no read-up' or 'Simple Security Property' (4). If information is uncategorised there is no 'need-to-know' requirement.

an insider is not allowed to write information to a file with a lower classification, so-called 'no write-down' process, *-property, or Confinement property (4); an insider can only write to files with a classification at least equal to or higher than the classification of the insider. It is not possible to write information without a category when the insider has a 'need-to-know' status to prevent information leaking from that category.

This is depicted in Figure 3.1.

The Bell-LaPadula model prevents information from flowing 'down' to parties through a hierarchy (vertical) but is not designed to prevent information from flowing in a horizontal fashion between people that might abuse the knowledge of information from one company combined with the combination of access to information from another company. Models such as the Chinese Wall Model (section 3.2) are designed for this purpose.

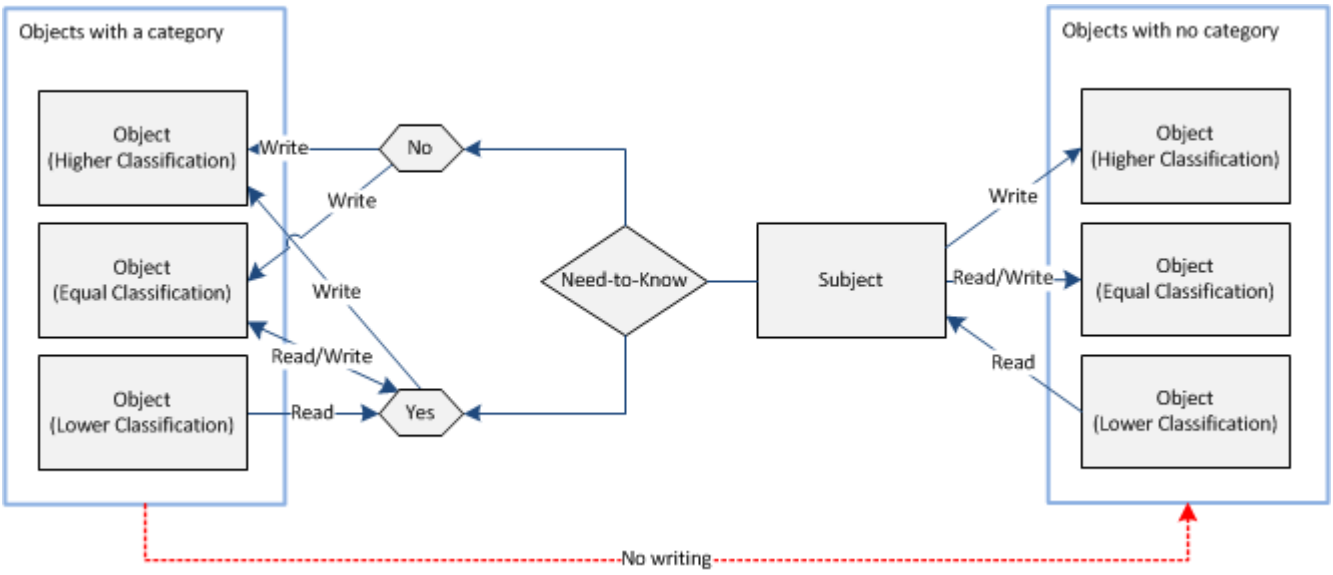


Figure 3.1: Bell-LaPadula model.

3.2 Chinese Wall Model.

The Chinese Wall Model (also known as the Brewer and Nash model) is a model which allows for dynamically changing access controls that protect against conflicts of interest (29). A conflict of interest is a set of conditions in which professional judgement concerning a primary interest (such as a patient's welfare or the validity of research) tends to be unduly influenced by a secondary interest (such as financial gain) (62).

The Chinese Wall Model is a multilateral model working on three levels of abstraction; Objects, Companies, and Groups of Non-Conflict (64) (see Figure 3.2).

An insider can only gain read information of a certain company when he never before read information of another company in another group. Other rules apply to writing information. an insider can only write information when he has access to the information of the company to which he wants to write, and should not have read access to information of any other company within another group where a conflict of interest may arise.

3.3 Jericho

According to their own statement[1] the `Jericho Forum is the leading international IT security thought-leadership association dedicated to advancing secure business in a global open-network environment. Members include top IT security officers from multi-national Fortune 500s and entrepreneurial user companies, major security vendors, government, and academics. Working together, members drive approaches and standards for a secure, collaborative online business world'. (..) 'The Jericho Forum is an international group of organizations working together to define and promote the solutions surrounding the issue of de-perimeterisation. They recognize that over the next few years, as technology and business continue to align closer to an open, Internet-driven world, the current security mechanisms that protect business information will not match the increasing demands for protection of business transactions and data in the future'.

The general idea behind the Jericho approach is that data should not be protected by defending the company perimeter (corporate network) by using defensive mechanisms such as firewalls, but rather that the data itself should be protected (56). This concept, dubbed 'deperimeterisation' moves away from the idea of a large defence

[1] https://www.opengroup.org/jericho/

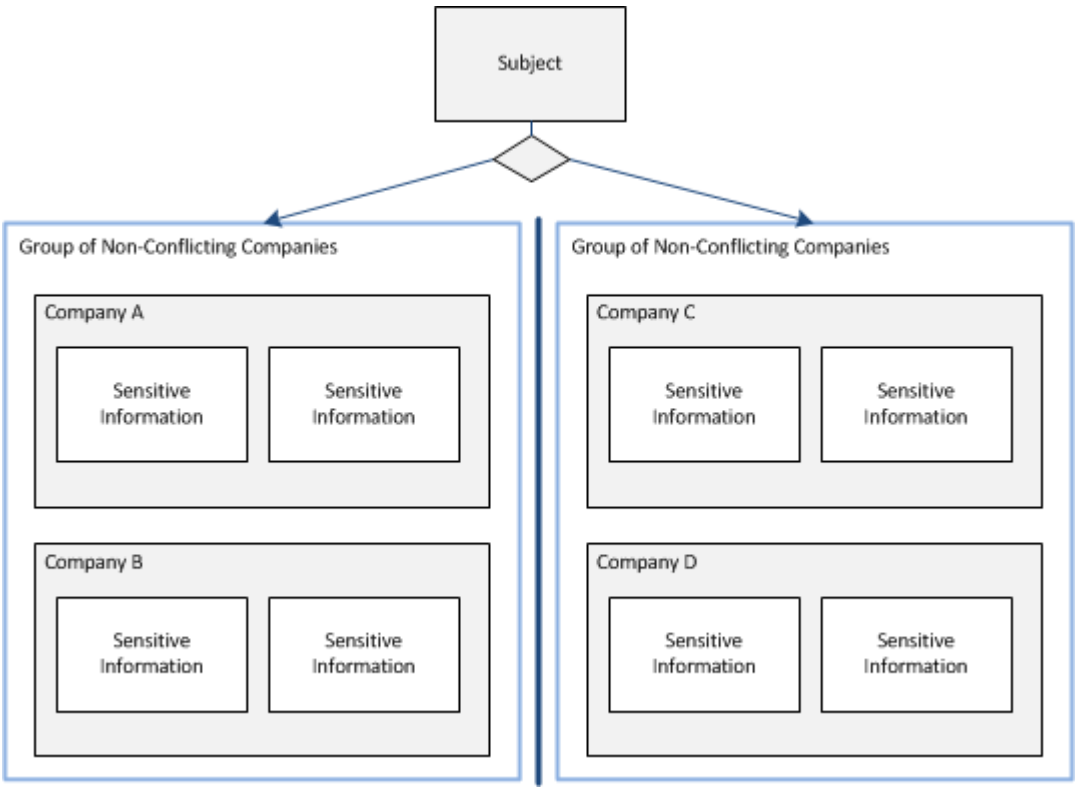


Figure 3.2: Chinese Wall model: an insider can have access to information within one group, but not to the other.

medium and proposes that each component handles its own security so that any file or machine can be placed within any (possibly insecure) network and still have a defensive perimeter to protect it.

Formulation 6. ***Deperimeterisation:*** *In a fully de-perimeterized network, every component will be independently secure, requiring systems and data protection on multiple levels, using a mixture of encryption, inherently secure communications, and data-level authentication (25).*

The stimulus for this approach is that `a broad consensus among computer scientists is emerging that the approach of patching and retrofitting networks, computing systems, and software to add security and reliability may be necessary in the short run but is inadequate for addressing the nation's cyber security needs' (13).

The Jericho ideas are presented in a document entitled `Jericho Commandments' (See Appendix A).

3.4 Relating Information Flow Models to a Covert Channel DLP model.

A covert channel is defined as a communication of sensitive information between two parties done via a secret channel that was not designed or intended for the sending of the information being sent, which violates the security policy (See also section 4.1 on the definition of Covert Channels).

An information flow model models an `ideal situation' in which information flows only over the defined channels. Such models but do not incorporate the possibility of covert channels; a piece of information is allowed to travel only between insiders with appropriate access rights, but information leakage for example due to the channel itself or the possibility of `hidden' information is not considered in information flow models.

The Jericho approach is somewhat different from the Bell-LaPadula model and the Chinese Wall model in that it is not really a model; it is a set of suggestions (in the `Jericho commandments') composed by an independent forum. The commandments encompass the storage of, access to, and security of information. The idea of de-perimeterisation is interesting. Instead of having a single point of network defence they suggest that each machine and piece of valuable information ought to be secure independently, and that it should only communicate with other secure machines. It is an approach that makes sense in environments where it gets increasingly more difficult to harden the company perimeter.

In our research we try to bridge the gap between theoretical information flow models, practical DLP solutions, and covert channel knowledge by creating a Covert Channel DLP model.

In our model we incorporate the idea of multi-level security; the DLP security policy (implementation of the security policy used to enforce it) that the DLP consults contains information regarding which insiders (do not) have access to what sensitive information and with whom they can share this information. Our model incorporates the scenarios of information retrieval from shared resources, sensitive information sharing between insiders, and (sensitive) information sharing with an outsider. In an actual implementation of the model the extensiveness of who has access to what information can be simple up to a very fine-grained control which allows the model to be workable in simple as well in complex situations.

The concepts of covert channels and Data Leakage Prevention are investigated further in chapter 4 (Covert Channels) and chapter 5 (Data Leakage Prevention).

Chapter 4

Covert Channels

This chapter is dedicated to Covert Channels. A look at several definitions of the term `covert channels' is given in section 4.1. The different types of covert channels and attacks are described in section 4.2. By `attacks' in the context of covert channels we mean the sending of sensitive information through a covert channel.

***

4.1 Covert Channels Defined.

The definition of `covert channels' varies depending on the consulted source. Because the topic of this thesis is covert channels it is important to define exactly what we mean by this term. Some definitions of covert channels:

The first definition was given by Lampson (36) in a paper on the problem of confining programs during their execution to prevent it from transmitting information to any other program except for its caller:

Formulation 7. *A communication channel is covert if it is neither designed nor intended to transfer information at all.*

From this definition it is clear that a covert channel differs from a regular channel in the way that the purpose of the channel was not to send this information.

In his book `A practical approach to identifying storage and timing channels' Kemmerer describes covert channels as (33):

Formulation 8. *Covert channels (..) use entities not normally viewed as data objects to transfer information from one insider to another.*

This definition does mention the hiding of the information being sent (`entities not normally viewed as data objects'; Channels which hide information (for instance, by putting data inside other information such as steganography) are not always considered to be covert channels. In this paper however, we consider them to be subliminal covert channels (See section 4.2).

Huskamp in `Covert communication channels in timesharing systems' (30) uses the following definition:

Formulation 9. *Those channels that are a result of resource allocation policies and resource management implementation.*

A note about this definition is made(46): The computing environment usually carries out resource allocation policies and implementation.

This definition does not mention anything the channel not being designed or intended for sending of information, but explicitly mentions resource allocation policies and resource management implementation.

The U.S. Department of Defence (DoD) has a definition in their 'Trusted Computer System Evaluation: The Orange Book' (46) which differs from the above definitions in that it mentions the use of a security policy:

Formulation 10. *A covert channel is any communication channel that can be exploited by a process to transfer information in a manner that violates the system's security policy.*

The security policy takes on a vital role in this definition. Within the DoD document sections 'Irrelevance of Discretionary Policy Models' and 'Dependency on Nondiscretionary Security Policy Models' it is claimed that a covert channel depends on non-discretionary security policy models and that the notion of covert channels is irrelevant to discretionary security models because discretionary models cannot prevent the release of sensitive information through legitimate program activity (which could be caused by a Trojan Horse acting on behalf of the user).

Formulation 11. *A communication channel is covert if it is based on transmission by storage into variables that describe resource states (24).*

This definition does not take into account any 'security policy' or the fact that a covert channel might be 'hidden'. Merely a channel that exists due to information being exchanged through resource state variables is mentioned./.

The Trusted Computer System Evaluation Criteria (TCSEC) standard defines a covert channel as follows:

Formulation 12. *Given a non-discretionary (e.g., mandatory) security policy model M and its interpretation I(M) in an operating system, any potential communication between two insiders I(Sh) and I(Si) of I(M) is covert if and only if any communication between the corresponding insiders Sh and Si of the model M is illegal in M.*

TCSEC's definition applies to communication between 'insiders' in which a distinct separation is made between the 'security policy model' and the interpretation of this model in an operating system, and it considers the communication covert when communication is illegal based on the security policy model as well as specific machine operational conditions. According to this definition network traffic is covert when the channel violates both the normal network protocol communications as well as the organizational policy.

We will give the following definition to covert channels;

Definition 4. Channels are considered covert channels when communication of sensitive information between two parties is done secretly via a channel that was not designed or intended for the sending of the information being sent which violates the security policy.

This definition considers sending sensitive information 'secretly' using otherwise legitimate channels as covert channels, but only when it violates the security policy. Sending 'non-sensitive' information or data which does not violate the security policy is not considered a covert channel. An assumption that is being made is that sensitive information sent over legitimate channels between two parties who should not be sharing this information is not allowed by the security policy. By using this definition we put great trust in a security policy's quality and extensiveness. For one, it should be clear what information is - and is not - allowed to be sent between different parties and over which channels. For the DLP security policy (implementation of the security policy) to be correct the security policy itself needs to be correct as well.

The notion of 'secret' is added to characterize the fact that the aim of a covert channel is to hide the communication. A subtle difference from encrypting traffic over legitimate channels and covert channels is that latter tries to hide the fact that communication is taking place, unlike encryption which does not camouflage the communication but only the information being sent. Forms of data hiding (such as Steganography in which data is hidden within other data such as pictures) are considered (subliminal) covert channels (See also section 4.2).

The next section shows the different categories followed by some covert channel examples.

4.2 Covert Channel Attacks.

There are two main categories of covert channels:

- Covert Storage Channels;
- Covert Timing Channels.

Besides these two main categories there are others mentioned in literature:

- Termination Channels;
- Resource Usage Channels;
- Power Channels;
- Subliminal Channels.

Storage Channels are channels based on (shared) data storage areas. According to (46) covert storage channels include all vehicles that would allow the direct or indirect writing of a storage location by one process and the direct or indirect reading of it by another. In the case of leaking sensitive information from an enterprise the 'process' could be read as an employee writing sensitive information to a shared resource and an outsider reading that information.

A Timing Channel is a covert channel in which one process signals information to another process by manipulation of the time it takes to perform a certain operation. The other process can observe this time to gain insight on what has happened. A definition by (46) is 'Covert timing channels include all vehicles that would allow one process to signal information to another process by modulating its own use of system resources in such a way that the change in response time observed by the second process would provide information'. In the case of leaking sensitive information an insider could manipulate the time in regular or seemingly uninteresting traffic to signal a message to the observing outsider.

A Termination channel is like a Timing channel in that it uses the factor 'time' to gain information. Instead of monitoring the time while a process is executing an operation it takes into account the time it takes a process to complete an operation.

Resource Usage/Power Channels rely on the manipulated difference in availability or usage of a certain resource (e.g., CPU usage or electricity usage). By monitoring these resources information might be derived indirectly.

A Subliminal Channel is a channel in which a message is hidden within another message. When such a message is intercepted it will raise no suspicion most of the time because the message itself is valid. Only when one knows what to look for the hidden message will become clear. Subliminal Channels are not always considered to be covert channels; the information is hidden within other data, the channel itself is not hidden. In this paper we do consider Subliminal Channels covert channels.

Under our definition of Covert Channels (Definition 4) there are several examples of covert channels. Of course, a requirement for these to actually be covert channels is that the behaviour is considered inadmissible with respect to the security policy.

- IP tunneling over DNS (Nstx);
- JitterBugs hardware keylogger (54);
- Unsecure shared network hard drive;
- Non-sensitive file attribute- or metadata manipulation to contain sensitive information (Subliminal channel);
- Public printer not configured with the 'secure printing' option enabled;
- Packet Morse code (Abusing packet delays to signal a message);
- Packet Signaling (Signaling information by sending different packets in a particular order,);
- Steganography (Subliminal channel);
- Using normally unused fields in network packets (Subliminal channel);

- Manipulating replies to requests (e.g. adjust the response send to an ICMP request (see section 4.2)).

IP tunnelling over DNS (or other forms of tunnelling) can be considered covert channels. The channel is used to retrieve other information than it was intended to be used for; Instead of DNS traffic IP traffic is transferred. Some firewalls use techniques to preserve the state of packets to prevent tunnelling, but it is an expensive process, since it is required that the firewall keeps the state of all protocols going through it (57).

A hardware keylogger is attached between the keyboard and the computer. All traffic between the two is saved in the keylogger memory which can later be retrieved. The keylogger can be viewed as a shared storage location for keyboard input that receives data from the keyboard, stores it, and passes it on to the computer. This attack is useful when an attacker has (short) physical access to the computer but is not able to log on to the computer or access any information on it directly (e.g., no login credentials).

Insecure shared network hard drives or any storage location that can be written to from within an organization and also be accessed by outsiders is a covert channel; sensitive information written to the storage resource can be read by an outsider which leads to data leakage.

Manipulation of file attributes and metadata can be used to embed information from sensitive files into non-sensitive files. Non-sensitive files can then be sent to outsiders who can retrieve the sensitive information within. The quantity of information that can be sent this way depends on the type of file; A picture file likely contains a lot more metadata (e.g., data the photo was taken, camera type, ISO level) than a plain `.txt' text file.

Public printers which are not secured (having to type a password at the printer for each print job) are covert channels because sensitive information can be sent to the printer and someone else who does not have the appropriate clearance can access that information. A secure printer with password ensures that only the person who initiated the print job can actually print the information.

Packet Morse code uses delays between packet sending to transport a message. An example of this would be sending a certain amount of DNS requests to an attacker-owned server each minute for a known domain. The attacker can filter the incoming DNS requests for the specific domain and see how many requests where done each minute and translate the number of requests per minute to information, such as a character. Figure 4.1 demonstrates sending the message `hello' (`h' equals two requests per minute, `e' three, `l' one, and `o' four) from an insider with IP 192.168.1.1 to an outsider owned DNS machine with IP 1.2.3.4 by making timed DNS requests for `www.netbsd.org'.

Because the traffic is often allowed by the security policy and passes the firewall unchanged this generally does not generate suspicion if a protocol is used that normally sends lots of separate packets every minute; twenty FTP connect requests might look strange and raise suspicion, but twenty HTTP requests do not.

Packet signalling uses different protocol packets to signal a message. An example is using combinations of HTTP, DHCP and FTP packets to signal a message. Because these packets blend into regular traffic and do not arouse suspicion the fact that communication is taking place by means of combining the different packet combinations remains hidden.

Steganography is the hiding of information within other information. A frequently used example of steganography is hiding a hidden message within a graphic file by changing the colour of every 100th pixel to correspond to a character.

Using normally unused fields in network packets can be used to send information by piggybacking along on allowed traffic.

An example of an attack over a storage channel is storing information in the IP, TCP, UDP, and ICMP Protocol Header Fields.

One such attack is known as the **ICMP Error Message Echoing Integrity Probe**. It can give an attacker a lot of information about a target's operating system. The technique is to get an ICMP Error message[1] from a target. This message is then be analysed; some implementations of the IP/ICMP stack alter the IP header when sending back ICMP error messages. Inspection of the protocol header fields can provide useful information (e.g., Operating System

[1] Port Unreachable, Destination Unreachable, Redirect, Source Quench, Time Exceeded, Parameter Problem

No.	Time	Source	Destination	Protocol	Info
1	2010-02-10 18:37:13.496046	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
2	2010-02-10 18:37:13.529802	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
3	2010-02-10 18:37:45.130387	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
4	2010-02-10 18:37:45.480926	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
5	2010-02-10 18:39:25.356625	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
6	2010-02-10 18:39:25.744529	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
7	2010-02-10 18:39:35.673243	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
8	2010-02-10 18:39:35.842232	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
9	2010-02-10 18:39:46.221463	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
10	2010-02-10 18:39:46.472285	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
11	2010-02-10 18:42:11.377429	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
12	2010-02-10 18:42:11.556383	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
13	2010-02-10 18:43:42.527289	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
14	2010-02-10 18:43:42.869043	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
15	2010-02-10 18:45:12.537282	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
16	2010-02-10 18:45:12.775521	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
17	2010-02-10 18:45:20.216782	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
18	2010-02-10 18:45:20.628789	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
19	2010-02-10 18:45:33.416824	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
20	2010-02-10 18:45:33.837921	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12
21	2010-02-10 18:45:50.209425	192.168.1.1	1.2.3.4	DNS	Standard query A www.netbsd.org
22	2010-02-10 18:45:50.572273	1.2.3.4	192.168.1.1	DNS	Standard query response A 204.152.190.12

Figure 4.1: Example dump for a packet Morse code covert channel.

on target). There are other possible probes that rely on differences in implementation of the RFCs; **FIN probes**, **ISN sampling**, **ACK value**, **Fragmentation handling**, and many more (39).

Steganography, file attribute- and metadata manipulation, and using normally unused fields in network packets are Subliminal Channels; the information itself is hidden, not the channel.

Timing information can be useful to gain information. Knowing that certain operations require a longer time to compute have proved to be an invaluable tool in cracking certain cryptographic schemes (see below). Making sure that operations take an equal amount of time regardless of the input can often fix such issues.

The code snippet below is an example of how a bad implementation can lead to information disclosure. The script contains several flaws; 1) Length check quits program immediately if the length is not correct, 2) Depending on the correctness of each character the program quits earlier. Measuring the amount of time it takes for the program to finish leads to the exposure of the password.

```
char[] passwd = "abcde";
char[] input = userinput();

pwOK = false;

if(input.length != passwd.length)
  return -1;

for(int i=0; i<passwd.length; i++)
  if(input[i] != passwd[i])
    return -1;

return 0;
```

Port knocking is the process of opening a port for communication by sending (different) requests to a certain series of ports. An example would be to 'knock' on ports 20, 31 and 53 by sending a series of packets, a process which would then allow the knocker to connect to port 23 (which would be locked otherwise). More sophisticated implementations of port knocking keep in mind the delay between the knocks and there are even port knocking implementations that allow single-port port knocking.

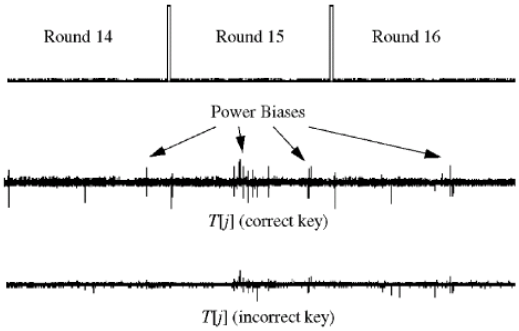


Figure 4.2: *Differential Power Analysis results for the last 3 DES rounds on a HC05-based smart card (source (40). The signal for the correct key has biases twice as large as the signal for the incorrect key.*

An example of data leakage through a power channel is a smartcard power analysis attack. In this attack the smartcard power consumptions is monitored while it is busy doing cryptographic computations. Due to fluctuations in the power consumption it is sometimes possible to retrieve the cryptographic key.

Note that the above smartcard power attack can be a side-channel attack which will not be considered in the model presented in this paper.

Some things that we do not consider in our model because they are either not covert channels or because they are out of scope;

- Side-channels;
- IP address collection through web surfing;
- Number of pizza deliveries to the Pentagon (which could indicate 'something big' is going on);
- Sending of encrypted information through legitimate channels;

A special class of a covert channel is a 'side-channel'. Side-channels refer to channels through which information leaks from a system because of the system's characteristics when in use (e.g., power consumption or time usage). Side-channel attacks have gained popularity due to their capabilities of finding flaws in cryptography implementations. Side-channels will not be considered in this paper; although they are covert channels we focus merely on intentional data leakage.

Chapter 5

Data Leakage Protection

Cases of leaked sensitive files, such as confidential enterprise reports or privacy-sensitive documents with customer- or staff information, have been in the news on numerous occasions. A confidential document wrongly attached to an e-mail instead of the intended document, faxes containing private information sent to the wrong fax number, and unprotected USB-sticks or laptops with secret documents lost on the street are only three examples.

According to the Global Data Leakage Report 2009 (31) there were 735 reported cases of data leakage in 2009 around the world. The 2010 version which reports 382 in the first half of 2010, 169 (44%) of such leaks reported to be intentional. These reports are based upon the leak database maintained by InfoWatch analytical. The database includes global data on any leaks reported by media, blogs, web forums, and any other public sources worldwide. Highly developed countries each suffer at least two leaks per year per million of population today. An average personal data leak encompasses approximately 750 thousand records.

While the consequences for companies can be severe in the form of breaking consumer protection laws (such as the Health Insurance Portability and Accountability Act (HIPAA), the Gramm-Leach-Bliley Act (GLBA), Sarbanes-Oxley (SOX), and the the Children's Internet Protection Act (CIPA) in the Unites States), money loss (or in the worst case, bankruptcy), or a bad reputation, the consequences for the victims are perhaps far worse. Cases of identity theft in which stolen personal information is abused to 'fake' someone's identity for financial profit are becoming increasingly more common. In some cases these crimes can destroy a person's life; giant debts, a negative profile, the inability to get new loans or credit cards, and even arrests occur. It has become paramount for companies to protect sensitive information.

***

5.1 DLP Defined.

To protect themselves from information leakage organizations employ DLP solutions in order to counter sensitive information from escaping the company network. DLP systems are used to gain insight in the 'who, how, why, and when' questions related to data usage and to enforce data usage policy to prevent data leakage, both accidental and malicious. A more detailed definition is given below:

Definition 5. Data Leakage Prevention (DLP) is a concept in which by defining rules and policies the data flow inside and outside the corporation can be controlled (44).

This definition states that a DLP solution uses a set of rules and policies that defines what data transfer is allowed between different parties and what is not. This information can already be present in the form of either company policy or technical security policy, but is sometimes also (partially) determined by the content and context of the information itself (e.g., by means of Data Content Analysis). DLP is not limited to information flow between parties within a

company but can also be used to prevent sensitive information flow to parties outside of the company. The word 'controlled' used in the definition is open to interpretation in relationship to 'broadness' or 'comprehensiveness'; No assumptions are made about what channels and protocols should be monitored or how granular or multi-layered these controls should be. Different types and approaches of DLP solutions differ in their way of achieving information loss control. In section 5.4 and 5.3 this is examined.

5.2 DLP Overview.

In a simple network users are connected to each other and the Internet through a single connection which is guarded by a firewall to block unwanted access from the outside (see Figure 5.1).

Most networks include more network- and Internet connected equipment and sub-networks and are more complex than the simple network of Figure 5.1; These networks include employees working at home who can connect to the company network via a VPN connection, laptop users who connect to the Internet and the internal network using Wi-Fi access points, have inter-connected servers, backup- and network storage equipment at different locations within the network, and also devices allow internet connectivity to devices such as printers, fax machines, telephones and PDA's (see Figure 5.2).

These networks are often protected from attackers from the outside by means of a firewall and/or Intrusion Detection/Prevention System (IDS/IPS). A firewall protects networked computers from intentional hostile intrusion that could compromise confidentiality or result in data corruption or denial of service (19). An intrusion detection system (IDS) inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system (19).

To prevent unwanted situations from occurring (such as in Figure 5.3) the network needs to be carefully constructed to prevent attackers from gaining access to the network via unsecured points of access. The connection of a company to the Internet is quite often regulated by a firewall. The firewall provides a centrally controlled connection to the Internet via which employees can connect to the Internet safely and which blocks outsiders who are not allowed to connect to the company network.

5.3 DLP Evolution.

Throughout time, Data Leakage Prevention has existed in several forms, but not always under the same name, using the same techniques, or even with the goal of preventing information leakage; even simple file-attributes disallowing

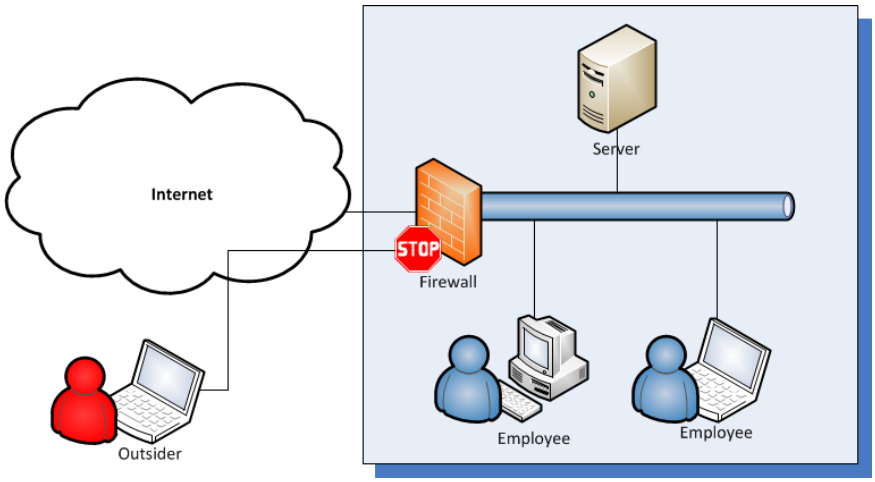


Figure 5.1: Enterprise Network: Simplified.

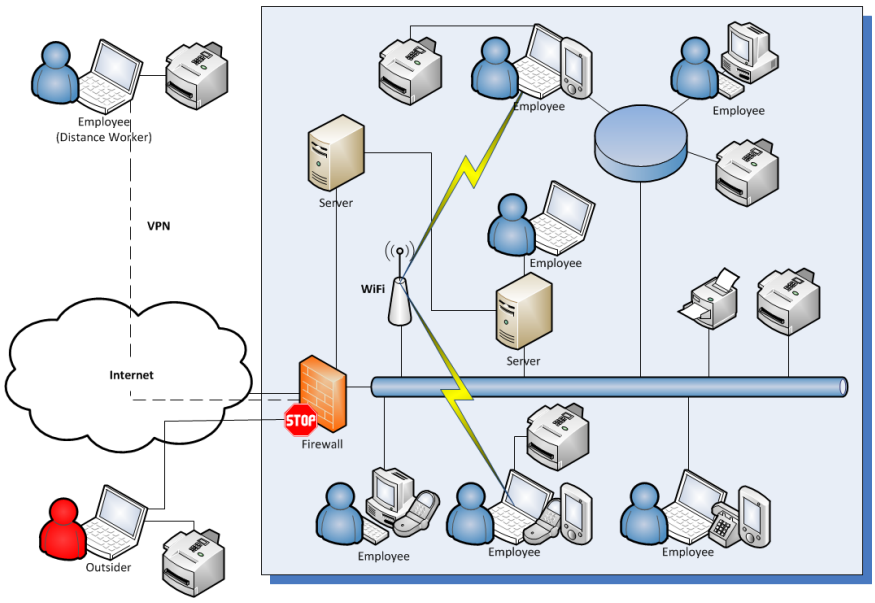


Figure 5.2: Enterprise Network: Extended.

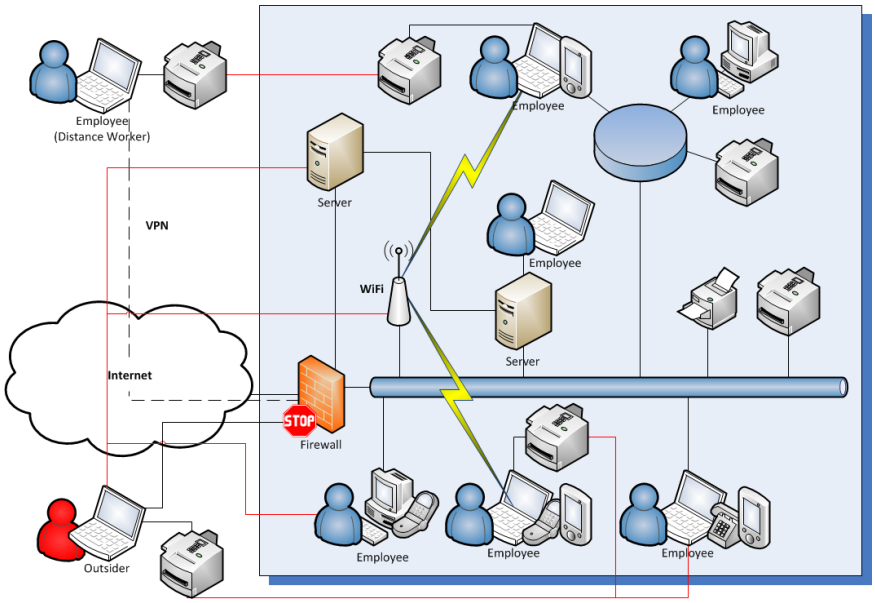


Figure 5.3: Enterprise Network: Unwanted connections (marked red).

everyone but the file owner to read and write the file could be considered to be a form of data leakage prevention. In this section we take a look at the history of DLP and get a glimpse into the future.

History.

The white paper '**The Evolution of DLP: Reducing Complexity**' describes the evolution of DLP (18). Before DLP would be a market on its own there were technologies designed for passively monitoring network activity. Due to new consumer protection legislation (among others events) these technologies evolved to provide sensitive information protection. The first solutions were network-based software technologies which were created to (only) detect the leakage of sensitive data from the corporate network by passively monitoring all outbound traffic. The next step was prevention. With the thought of overloaded servers possibly allowing data to leak outside the protected network, most early DLP vendors used a modular architecture to allow them to offload additional processing requirements

to servers running additional components, ensuring ongoing and effective detection of sensitive data. Typical DLP solutions nowadays use separate servers for separate components (e.g., separate servers for e-mail blocking, management, web blocking).

According to Oltsik, the most common DLP implementation around 2005 was as a network gateway for filtering Application Layer content, a tactical point tool for blocking content on the network (47).

Future.

Oltsik mentions that `*DLP has an identity crisis*' (47) and that it has evolved into:

- An architecture. Network DLP gateways, desktop software, and file systems agents are now part of a broader network architecture with central command-and-control and policy management;
- An integration nexus. DLP now integrates with encryption software, virtual desktop technology, and eRM;
- A policy engine. "Canned" compliance policies are no longer enough for large organizations. They want to develop and test custom policies for their own internal content. This is especially true for high security organizations or those with lots of digital intellectual property.
- A meta data hub. DLP is getting better at discovering and classifying data. More recently, DLP is gaining knowledge on who is actually using the data as well.

DLP solutions have morphed from passive network-activity monitoring technologies to comprehensive enterprise-wide data governance instruments.

According to DLP Experts LLC the future of DLP will be a unified DLP architecture which, in contrast to the modular architecture, combines all components into a single appliance, integrated with existing infrastructure (18). This single, multi-function DLP appliance will replace the high complexity multi-server modular architecture of `classical DLP solutions', according to the authors.

5.4 Types of DLP solutions.

Within DLP there are two classical approaches: Network DLP and Host-based DLP. The first approach focuses on information that flows over the network between insiders and to / from outsiders. The latter approach focuses on the information at the end-points and is typically installed on a computer system.

Network DLP.

Network DLP systems frequently consist of dedicated hardware- or software solutions within the internal network to detect unauthorized streams of data. Network DLP systems often filter traffic on the Application Layer (Layer 7) including protocols such as DNS, FTP, HTTP, IMAP, POP, and SMTP. An advantage is that a Network DLP system is easy to install and has a relatively low cost of ownership. A disadvantage is that they cannot prevent users from copying sensitive information to other devices physically connected to a computer, such as USB-sticks.

Host-based DLP.

Host-based systems are often built-up by workstations (and servers) equipped with DLP software. These systems focus on the leaking of information on a host-based scale; copy/pasting information from a sensitive document to an e-mail message, printing sensitive documents, copying to USB, making screenshots, and more actions which are hard (or impossible) to detect when merely monitoring the network. An advantage of Host-based DLP solutions is that they are able to check whether sensitive information is transferred to physical devices (including smartphones and USB-sticks). A disadvantage is that the DLP software must be installed on every workstation within the network;

sometimes this might not even be possible (e.g., in the case of smartphones which cannot run all software) or might not be practical due to the nature or volume of workstations.

Besides these two main categories there is much difference between extensiveness regarding what data / information to monitor and what channels or protocols to filter. Some DLP solutions stay very basic while others are more extensive and include features such as dynamic data classification, support for encrypted protocols, and have self-training possibilities (Also see section 5.3).

5.5 The D in DLP.

The name DLP, or *Data* Leakage Prevention, implies that such solutions are responsible for the protection of 'data' within an organization. Yet, there are also synonyms for DLP such as Information Leak Prevention (ILP), Information Leak Detection and Prevention (ILDP), and Information Protection and Control (IPC) which suggest that 'information' is protected (See chapter 2 for the distinction between data and information). In this section we take a look at the different types of data that exist and at the different techniques that can be used to find (hidden) information within this data.

As stated in section 5.4 there are different types of DLP solutions (Network, Host-based). Depending on the DLP solution the information is protected at different points in a communication. Besides the different points of protection there are also different types of data to protect; Data in motion, data at rest, and data in use (see below). In our model, the three stages generally apply to (sensitive) information instead of data.

Data in motion.

Data in motion (or data in transit) is data travelling from one location to another over a network. This can be data from one person's computer to another person's computer (e.g., e-mail or instant messages), but also traffic to, from, and between servers, storage locations, routers/switches, and other devices such as USB-sticks or smartphone.

Network DLP systems monitor data flowing through the corporate network and can thus be qualified as monitoring 'data in motion'. Data being copied to locally attached devices are not monitored by such Network DLP systems, but can be filtered by Host-based DLP solutions.

Data at rest.

Data at rest is data that is stored (e.g., documents stored on the hard drive of workstations or backup data on tape).

DLP systems enable enterprises to search within the internal network to identify risks related to data that is stored and warn about detected data at locations where certain data should not be stored.

Data in use.

Data in use is data on which employees are currently performing work or to which data is currently being written (e.g., in case of log-files). Host-based DLP solutions can monitor this kind of data, as well as copies being made to locally attached devices such as USB-sticks, but can also prevent it from entering the network by blocking such traffic.

No matter whether the data is in motion, at rest, or in use, to successfully prevent information leakage it is important that the potential sensitive information within what appears to be data or non-sensitive information can be found. Section 5.6 takes a look at Deep Content Analysis which is used to detect sensitive information in data.

5.6 Detection Techniques

In order to be able to prevent or detect information leakage the DLP system generally needs to have the functionality to detect information in data on networks, within network data streams and in traffic to computer peripherals (e.g., printers, fax machines). According to an informal survey of major DLP vendors, content discovery is a part of 50-60% of initial DLP purchases, with deployment within the first 12 months of implementation (42).

Deep Content Analysis (DCA) techniques filter data according to a certain policy, which defines what (types of) content to protect.

Data analysis is done on two domains; 'Content' and 'Context'. The content of a file involves analysis of the actual file itself (up to bit-level analysis). The context of a file is divided in 'Meta-data' and 'Business Context'. Meta-data are linked directly to the file and includes attributes such as the file owner, the file size, date of creation, day of last access, etcetera. The Business Context is information not linked directly to the file itself but the context surrounding the file. Examples includes information on the system a file is stored on, what applications access the file, the time of day when the file was accessed, sender and recipients in the case of e-mails, etcetera.

Who handles the information also matters; employees in certain roles (such as database administrators) are allowed to use certain information in other ways than employees in other roles (such as finance).

There are generally two types of data that is searched for in Deep Content Analysis; 'registered data' and 'described data'. Registered data is data classified by detecting an exact occurrence of known data. Described data is data that is identified by detecting certain patterns within the scanned data. Registered data finding can be useful for detecting information known to be sensitive such as specific phrases, numbers, or bit-sequences. Described data is more useful when trying to detect certain types of information, not limited to known data. Examples include (non-specific) social security numbers, credit card numbers, street addresses, and other information which can be identified because it follows a certain pattern. Described data can result in 'false positives' when it detects a pattern in information when in reality there is no pattern.

DCA techniques can be used to detect (parts of) information within data at rest, data in motion, and data in use (43). When a sensitive piece of information is found leaving the company the DLP system triggers the appropriate alert and action to be taken. Additionally, these techniques can be used to (partially) automate the classification process of information (determining which information is 'sensitive').

Traditional DCA techniques

Traditional DCA techniques include rule-based/regular expressions, database fingerprinting, exact file matching, partial document matching, statistical analysis, conceptual/lexicon analyses, and category matching (43).

Rule-based DCA techniques and regular expressions are used to find information that follows a registered or simple described pattern. Rule-based matches the information with a set of rules to see if any of them are triggered. A simple example of rule-based DCA is keyword matching (e.g., the word 'Confidential' is found within a document) or rules for scanning certain data (streams) for regular expressions.

As an example, the FortiGate DLP system includes a rule called 'Email-US-SSN' which examines E-mail traffic (protocols SMTP, IMAP, and POP3) for U.S. Social Security numbers by matching the 'body' field contents with the regular expression (20):

```
\b(?!000)([0-6]\d{2}|7([0-6]\d|7[012]))([ -]?)(?!00)\d\d\3(?!0000)\d{4}(\b|\W)
```

This expression looks for a number in the range 001 – 772, followed by a separator (a blank space, or a '-', or nothing), then a number in the range 01 – 99, then the same separator again, and lastly a number in the range 0001 – 9999.

```
----
Date: Sat, 1 Jan 2000 00:00:00 -0600 (CST)
```

```
From: maliciousinsider@company.net
To: outsider@world.net
MIME-Version: 1.0
Content-Type: TEXT/PLAIN; charset=US-ASCII

List of secret SSN:
123-45-6789
234 56 7890
345678901
001 01 0001
772 99 9999
----
```

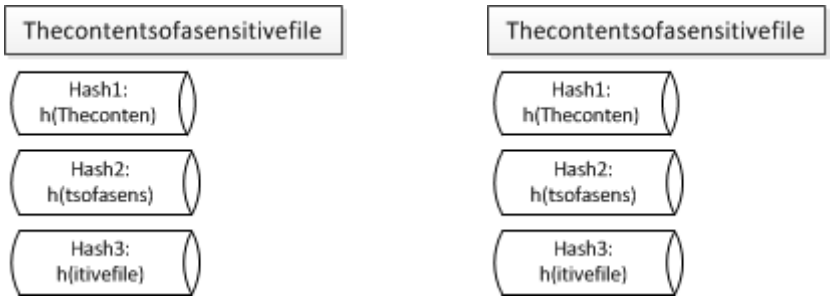
Some more examples of regular expressions for detecting credit card numbers and social security numbers can be found at the website of the open-source DLP solution OpenDLP[1].

These easy techniques work best on information with a simple structure for which the pattern or the exact contents is known beforehand, but fails horribly when this is not known. Another downside is the chance of false positives when using regular expressions (e.g., in the above example a mail containing `*Call me at 555-55-5555*' would be flagged as well).

Database fingerprinting is used when detection of linked information is required. Sensitive information examples are loaded from a database and matched with the information to be examined. Data from multiple fields can be used to form combinations of information. The fields could be registered data or described data. For instance, a credit card number used on a sales website would be no information leakage if it is the credit card number of the employee visiting the website.

Exact file matching is a simple detection mechanism in which hashes of files are matched with a blacklist of hashes of files that should not be sent (and/or a whitelist of files that are allowed to be sent), and which trigger the DLP system when an attempt is made to transfer sensitive files. This technique works on all types of files, also on files for which textual analysis would not always be possible (e.g., audio files or binaries). Another advantage is the fact that there are practically no false positives (given large enough hashes to avoid collision). The problem with this technique is that with little alteration (file size, file attributes, a bit of the contents) the file will get through this filter.

Partial document matching is more sophisticated than exact file matching; even when pieces of content are altered the information leakage will still be detected. A basic technique to accomplish this is hashing parts of the document instead of making a hash of the whole file (Illustrated in Figure 5.4a). There can also be an 'overlap' between the hashed contents (Illustrated in Figure 5.4b). This process is called 'Cyclical Hashing'.



(a) Non-overlapping hashing (b) Overlapping hashing

Figure 5.4: Partial document matching

Statistical analysis is a technique which uses the probability that an examined piece of data is sensitive information. Although the techniques differ the general approach is that knowledge of a large collection of known sensitive information is used to determine whether or not a new piece of information is likely to be sensitive information. Two methods that can be used are frequentest statistics and Bayesian inference. Learning algorithms (see next

[1] Regular Expressions OpenDLP: http://code.google.com/p/opendlp/wiki/RegularExpressions

section) often employ statistical analysis methods to learn about unknown information. Although statistical analysis can successfully be used for unstructured information, it is also prone to both false positives (a non-sensitive piece of information is flagged as sensitive) and false negatives (a sensitive piece of information is labelled non-sensitive).

Conceptual/Lexicon analysis uses techniques such as combinations of phrases and words and their proximity to detect sensitive information or unwanted behaviour. It can be used in more complex structured data which cannot be easily described using regular expressions or simple rules. An example that is commonly given is that this analysis technique can be used to detect (possibly) inappropriate behaviour such as job hunting on the job by monitoring keywords on web pages an employee visits (e.g., 'job', 'find a new career', 'monsterboard', 'curriculum vitae') in a certain timespan.

Category matching examines data for regulatory compliance to categorize it. These pre-built categories have their own rules for common types of information (e.g., credit card numbers) and can be used for information that is created according to the guidelines that describe the information within the category. Examples of predefined categories are HIPAA, GLBA, and PCI DSS (Health Insurance Portability and Accountability Act, GrammLeachBliley Act, Payment Card Industry Data Security Standard). The advantage is that these categories are easy to configure saving a lot of time that would be wasted configuring your own policies from scratch. The downside is that information which does not follow the rules of the category exactly is not always detected.

New DCA techniques.

Searching in encrypted data

Brinkman explains three methods for searching in encrypted data in his PhD thesis (**?**); using a trapdoor mechanism using encrypted XML to search for occurrences of words, secret sharing in which the data, the question, and the answers are represented as shared polynomials, or homomorphic encryption functions that allows for simple operations directly on the encrypted information.

Searching in encrypted data is useful to detect the transfer of sensitive information on encrypted channels. In Brinkman's paper, the server on which the information is stored is the untrusted party. In a company, the insiders leaking information and outsiders accessing the information are the untrusted parties.

Profiling

A technique that can prove to be effective is 'profiling'; using behavioural patterns to create a 'profile' from which information about habits and preferences can be deducted. The process of creating a profile and using the information to gain usable knowledge from it has been used already in the fields of commerce (e.g., targeted advertisements, suggestions for products) and fraud prevention. Besides these, profiling can also be used as a tracking practice in which, for example, the (online) behaviour of employees is monitored.

Profiling practices can range from simple logging to complicated machine learning practices in which many parameters are considered in creating a profile. For fraud detection and other uses of profiling which has the goal to detect oddities anomaly detection plays a great role. The profile is then used as a template for normal behaviour.

With regards to Covert Channel Data Leakage Protection the detection of information leakage and covert channels we see two options; using profiling techniques on the persons using sensitive information, or on the channels used to transfer the information.

In the first option a profile of the information a person uses can prove to be an effective method of information leakage detection; if anomalies in information requests appear (e.g., a person working at one department suddenly requesting documents of an entirely different department) this can be an indication of a possible information leakage incident. A sudden increase of traffic from an individual is another indication. On their own, these observations appear to be of less interest and would likely not raise a lot of suspicion. By building a profile which combines these observations it is possible to make a better judgement regarding a possible information leakage event instead of a judgement based on a single observation (as is often the case with for example log files). Creating profiles of persons can potentially conflict with (inter)national privacy laws.

Another option is creating a profile of channels. Possible information that might be included in a profile is;

- The times at which a channel is used;
- The traffic volume;
- The protocol used to transfer information on the channel;
- Encryption of information on the channel.

Timing information can be used to detect possible timing channels. The list of possible parameters to include in a profile can be dependent on the channels which are monitored. The idea of profiling communication channels has already been the insider of research (22), (23).

Machine Learning

The traditional DCA techniques rely either on registered data or on described data. The trouble is that the first requires collecting a lot of sensitive documents that need to be protected and the latter involves the time consuming and difficult process of creating regular expressions and rules to identify unknown sensitive documents. An alternative is using machine learning algorithms and artificial intelligence methods that allow systems to learn to distinguish sensitive from non-sensitive information (`classification').

The two major directions within machine learning are algorithms that use `supervised learning' and those that use `unsupervised learning'.

When training a supervised learning algorithm the training data consists of a certain input with a known desired output value. Advantages of this method are that complex patterns can be learned and that the performance is generally good. When classifying it allows the supervisor to predetermine the different classes. The downside is that the supervisor has to label the desired output value for each input.

Unsupervised learning differs from supervised learning in that the training data is unlabelled, meaning that for a certain input the desired output value is not given (and sometimes not even known). The advantage is that it is not necessary to take time to create categories for output, and that it is possible to find interesting patterns in data which were not known beforehand ('data mining'). The downside is that due to the absence of creating manual categories the algorithm might make other categories than intended.

An example of a product using a (supervised) learning algorithm is Symantec's DLP 11. It employs Vector Machine Learning (VML) which the manufacturer describe as `*a new detection technology for unstructured data that learns how to recognize sensitive data based on samples. Creating policies using VML to protect unstructured data like source code, product formulas and other intellectual property is more accurate than describing the data, and is less time consuming than fingerprinting all your sensitive data*'[2]

The technique that they used is the concept of Support Vector Machines (SVM). An SVM represents training examples as points in (high dimension) space and separates them as wide as possible (maximum margin) by a hyper plane. New and unknown examples are represented in the same space and classified to the category that applies to points in that location.

PoC: ICMP Covert Channel Detection

To look at the potential of these new detection methods a Proof of Concept has been performed on ICMP ('ping') traffic. The goal of this experiment was to test if learning algorithms are able to detect regular packets from changed packets within this traffic (such packets could be an indication of a covert channel). A learning method called `Artificial Neural Networks' was used.

Neural Networks background
Artificial neural networks are simulated biological neural networks; a neural network, such as the brain, is composed

[2]http://phoenix-es.co.uk/news/Symantec-DLP-11-Now-available.aspx

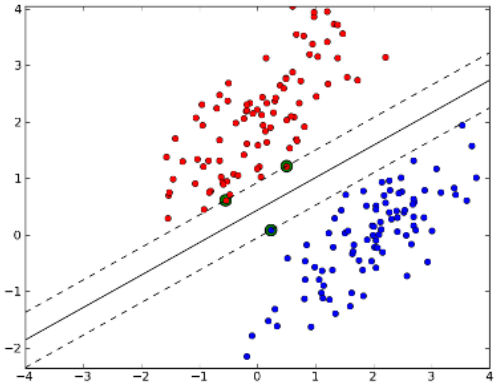


Figure 5.5: Example of a linear SVM (Source: mblondel.org)

of interconnected neurons which communicate using electrochemical signals. Each neuron has a number of inputs and outputs. The outputs of one neuron are connected to the input of another through synapses. Depending on the input strength a neuron may send a signal; If a certain threshold is reached the neuron 'fires'. The strength of the signal it sends depends on the synapse strength.

An artificial neural network uses artificial neurons; each neuron has a number of inputs (with a certain strength/weight) and a threshold value. The threshold value is subtracted from the combined weight of the input and depending on the output value a signal is send. A difference with biological neurons is that an artificial neuron can also 'partially fire'; the signal it outputs is then weaker or different, unlike the biological equivalent which either fires or doesn't fire.

An artificial neural network can be trained on a set of instances with known input and output values. The artificial neurons change their thresholds to reach an as low as possible error rating. This trained neural network can then be used to 'predict' the output of a decision process on new input for which the output is not known.

Experiment
In our experiment we use artificial neural networks to determine if an ICMP ('ping') packet is either a regular packet or if it was changed before sending. In our experiment we send standard Ping packets and custom packets. In total 1363 packets were used (of which 241 were changed). The source code can be found in Appendix B.

The data input files were the numerical values of the 19 fields used in an ICMP packet (see 'Layout of an ICMP packet' and 'Data input files snippet' below).

Two changes were made to the input data;

- Instead of the regular hexadecimal values decimal values were used because the hexadecimal values could not be stored in the used 'double' variables. Instead of 'normal' conversion from hexadecimal to decimal the choice was made to simply 'replace' the characters with numbers (A/1, B/2, .. F/6); artificial neural networks are useful in pattern recognition scenarios. Due to this fact it makes sense to replace unsupported characters with supported ones to leave as much of the pattern intact as possible.
- Due to no size constraints the data field can be too large for the double int variable to hold. This was solved by simply truncating the data after 8 bytes. Another possible option here would have been using more variables to hold the whole message.

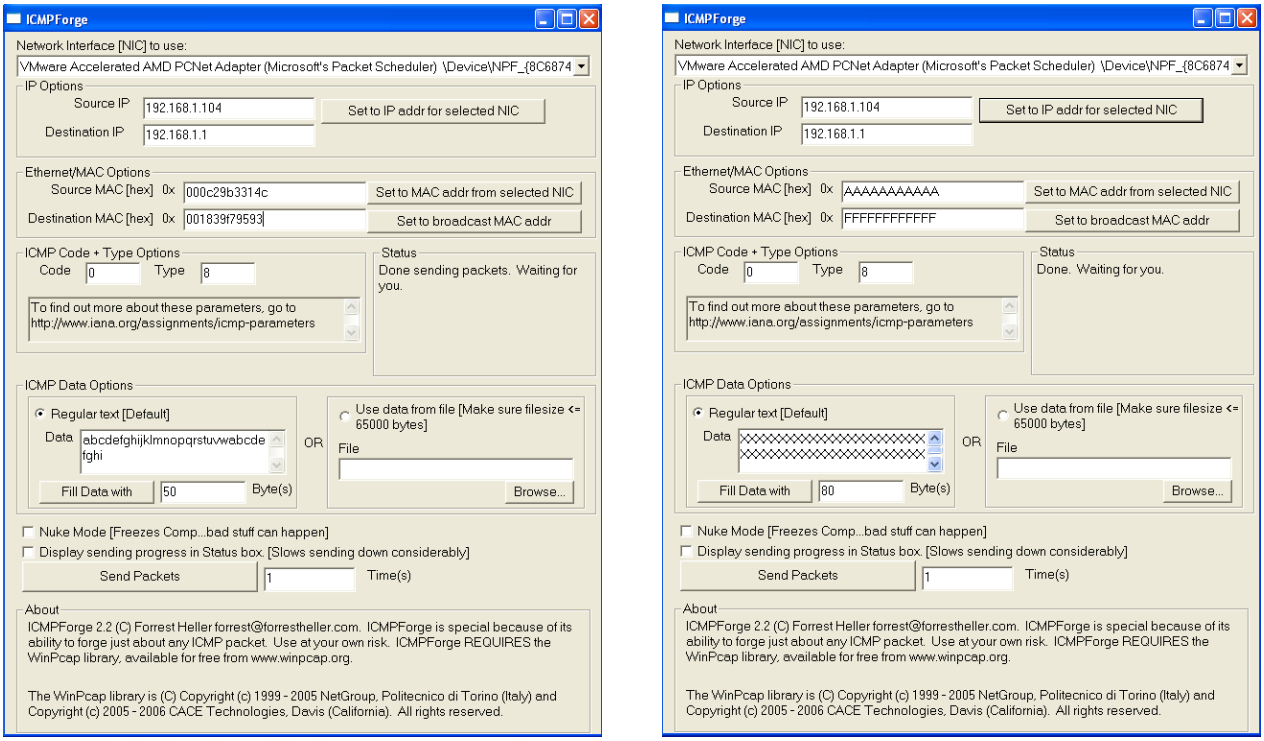


Figure 5.6: ICMPForge: A regular packet (left) and a changed packet (right)

Layout of an ICMP packet:

ETHERNET II:
6 bytes: Destination MAC (ff:ff:ff:ff:ff:ff)
6 bytes: Source MAC (aa:aa:aa:aa:aa:aa)
2 bytes: Type, IP (0x0800)

IP:
1 byte : Version (4, 45)
1 byte : Differentiated Services Codepoint (Default, ECN (Explicit Congestion Notification) 0x00)
2 bytes: Total Length
2 bytes: Identification
2 bytes: Fragment Offset (typically 00:00)
1 byte : Time To Live
1 byte : Protocol (01, ICMP)
2 bytes: Header checksum
4 bytes: Source IP
4 bytes: Destination IP

ICMP:
1 byte : ICMP type 08 (Echo (ping) request)
1 byte : Code 0
2 bytes: Checksum
2 bytes: Identifier
2 bytes: Sequence number

DATA: x bytes: data (normal ping 32 bytes: abcdefghijklmnopqrstuvwabcdefghi
(61:62:63:64:65:66:67:68:69:6a:6b:6c:6d:6e:6f:70:71:72:73:74:75:76:77:61:62:63:64:65:66:67:68:69))

Data input files snippet:

```
1363 19 1 (#inputs, #fields, output (1 regular packet, -1 non-regular packet)
001839679593 000329233143 0800 45 00 0033 2464 0000 80 01 8913 30180168 30180101 08 00 4053 <cutoff>
0200 0200 616263646566676
1
001839679593 000329233143 0800 45 00 0033 2464 0000 80 01 8913 30180168 30180101 08 00 4053 <cutoff>
0200 0200 616263646566676
1
....
666666666606 000329233104 0800 45 00 0051 5935 0000 66 01 4523 30180168 30180101 08 00 5444 <cutoff>
7565 3046 0038377934776975
-1
```

Results

The results of this experiment vary in success. Every time the network is trained the network is slightly different. This difference in networks gives different results when testing unknown input. For example, the following two networks were trained with a threshold of 1.5% (and 25 hidden neurons):

Network one training:

```
Max epochs   500000. Desired error: 0.0149999997.
Epochs            1. Current error: 0.2912335396. Bit fail 1363.
Epochs           50. Current error: 0.0944771171. Bit fail 145.
Epochs          100. Current error: 0.0542171523. Bit fail 121.
Epochs          150. Current error: 0.0447283611. Bit fail 47.
Epochs          200. Current error: 0.0272979569. Bit fail 43.
Epochs          250. Current error: 0.0269951914. Bit fail 42.
Epochs          300. Current error: 0.0451557487. Bit fail 42.
Epochs          350. Current error: 0.0235579554. Bit fail 36.
Epochs          400. Current error: 0.0222326461. Bit fail 32.
Epochs          450. Current error: 0.0232343692. Bit fail 32.
Epochs          460. Current error: 0.0132299848. Bit fail 16.
```

Network two training:

```
Max epochs   500000. Desired error: 0.0149999997.
Epochs            1. Current error: 0.2208992392. Bit fail 1363.
Epochs           50. Current error: 0.0645600930. Bit fail 145.
Epochs          100. Current error: 0.0236400422. Bit fail 32.
Epochs          150. Current error: 0.6609779000. Bit fail 1132.
Epochs          158. Current error: 0.0125349611. Bit fail 16.
```

As can be seen from the above listings it took the first around three times the amount of time to reach an error rate below 0.015.

The above two networks were used on two packets which were not in the training set. One was a genuine packet (first), another one was send using ICMPForge. The results for both networks on the same test data:

```
Unknown packet 1:
001839679593 000329233143 0800 45 00 0033 2465 0000 80 01 8912 30180168 30180101 08 00 3653 0200 <cut>
1300 616263646566676

Unknown packet 2:
002695004201 000329233104 0800 45 00 0039 2543 0000 66 01 1237 30180168 30180101 08 00 2256 7964 <cut>
6108 0073656564696564
```

The packets were given as input to both networks. A packet score closer to 1.0 indicates its likelihood to be a 'real packet', while a score close to -1.0 indicates a 'fake packet'.

Network one test run:

```
Real packet score: -0.809750, should be  1.000000, difference=1.809750
Fake packet score: -0.809750, should be -1.000000, difference=0.190250
```

Network two test run:

```
Real packet score: -0.768961, should be 1.000000, difference=1.768961
Fake packet score: -0.862363, should be -1.000000, difference=0.137637
```

What is interesting about these results is that for the first network the score for both packets is -0.809750. For both networks, the score for a genuine ping packet is extremely low compared to the expected value of 1. Determining fake packets goes fairly well; This is strange because due to the similarity of regular packets it was expected that regular packets would be easier to detect as such than it was to detect fake packets.

There are several options to tweak in the implementation.

The input parameter is the number of input neurons. This is generally configured at the number of input fields used. Since each packet has 19 fields it makes sense to keep this at 19.

The output is the expected output for each input set. Since we only want to know whether it is -1 (fake) or 1 (real) we keep this at 1.

As the num_layers variable suggests there is the possibility to add more than just the default two (input and output) neuron layers. In a 'normal' setup of an artificial neural network a third layer, called the 'hidden layer' is placed between the input and output layer. This hidden layer contains the 'hidden neurons'; neurons from which the activity is determined by the input and the weights on the connections between the input and the hidden units. The output activity is not directly influenced by the input layer, but by the activity of the hidden layer and the weights of the connections between the hidden layer and the output layer (see Figure 5.7).

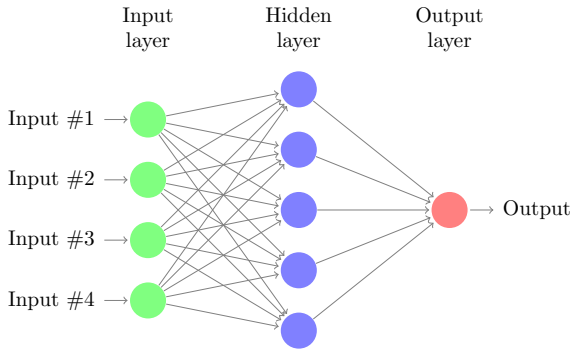


Figure 5.7: Neural network with three layers
(http://www.texample.net/tikz/examples/neural-network/).

Changing the number of hidden neurons from 25 to 30 hardly affects the results (error 1.5%):

```
Max epochs   500000. Desired error: 0.0149999997.
Epochs            1. Current error: 0.2836264670. Bit fail 1363.
Epochs           50. Current error: 0.0620035864. Bit fail 89.
Epochs          100. Current error: 0.0314841308. Bit fail 47.
Epochs          150. Current error: 0.0359490402. Bit fail 49.
Epochs          200. Current error: 0.0374101140. Bit fail 52.
```

```
...
Epochs         2850. Current error: 0.0330588035. Bit fail 42.
Epochs         2900. Current error: 0.0306430180. Bit fail 42.
Epochs         2950. Current error: 0.0243044458. Bit fail 32.
Epochs         3000. Current error: 0.0248717330. Bit fail 32.
Epochs         3050. Current error: 0.0171189010. Bit fail 23.
Epochs         3064. Current error: 0.0129910232. Bit fail 16.


Real packet score: -0.759441, should be  1.000000, difference=1.759441
Fake packet score: -0.759441, should be -1.000000, difference=0.240559
```

The real packet detection score difference remained roughly the same (still bad) while the fake packet score difference increased slightly.

The last parameter to change is the desired error score. Increasing this from 0.015 to 0.03 has a positive influence on the results:

```
Max epochs   500000. Desired error: 0.0299999993.
Epochs            1. Current error: 0.2227389961. Bit fail 1363.
Epochs           50. Current error: 0.0476080142. Bit fail 145.
Epochs           58. Current error: 0.0292909108. Bit fail 32.


Real packet score:  0.875820, should be  1.000000, difference=0.124180
Fake packet score: -0.700641, should be -1.000000, difference=0.299359
```

Although the fake packet score is slightly higher the results are more like one would expect; the real packets are quite similar and should be easy to detect, while fake packets can be completely different and are easy to detect (the fake packets that pretend to be real packets are expected to be harder to detect).

Conclusion

The trouble with learning methods such as neural networks is that 'debugging' them is very difficult. Tweaking and tuning the parameters is an important factor for successful learning. The experiment shows that it is possible to distinguish real and fake packets.

One possible conclusion for the need of reducing the desired error score parameter is overfitting (also referred to as overtraining); To reach the desired error of 0.015 the learning algorithm tweaks the neuron's weights in such a way that only the exact same packets are matched while it has no idea what to do with other packets it does not know. This makes the algorithm more accurate in matching known data but less accurate in matching new data, explaining the bad score.

To conclude whether this technique is workable in a real world scenario is difficult to determine solely based on this experiment; this experiment shows that, given the right parameters, it is possible to distinguish one type of network traffic ('regular' ICMP ping packets) from irregular traffic of the same type. However, network traffic in a real organisation consists of more dynamic traffic than ICMP 'ping' traffic which might be a lot harder to classify.

The threats to applying neural networks (and any other classification algorithm) are false positives and false negatives (classifying regular traffic as irregular traffic, resp. classifying irregular traffic as regular traffic). Techniques such as cost-sensitive learning (using classification costs, and base classification on 'minimum expected misclassification cost' or re-weighting training instances) can be used to reduce these. Utilizing other covert channel detection techniques (such as regular expressions or keyword mapping) in combination with learning methods can further decrease false classifications.

Chapter 6

Model.

The detection and prevention of information leakage in itself is a difficult problem. The detection and prevention of information leakage through covert channels poses an even greater challenge. By using the knowledge gained in the previous chapters regarding covert channels, DLP, and Information Flow models we will create a Covert Channel DLP model. This model focuses on intentional data leakage by insiders; deliberate actions taken by persons within an organization with the intention to 'smuggle' sensitive information out of the organization.

Section 6.1 gives an insight of the ideas behind the reasoning for our models (one DLP model, and one CC DLP model) and a short overview of the models. Section 6.2 presents a more detailed description of our Data Leakage model, followed by the Covert Channel model in section 6.3.

***

6.1 Overview.

Background
In order to create a model for a DLP model the first steps are finding out where the communication channels are, how to handle transfer of information in certain situations (between insiders, outsiders, and shared resources), and how to include a DLP system in these transfers. To go a bit deeper and create a covert channel DLP model requires additional information such as where covert channels can exist, how they can be detected, and how their existence can be prevented from existing at all or limited as much as possible. In the end, both the DLP model and the covert channel DLP model should allow for a situation where sensitive information can be transferred securely between authorized individuals and prevent unwanted situations from occurring.

The unwanted situations that we consider to allow for (covert channel) data leakage are:

1. A malicious insider directly sends - or signals - information to an outsider.
   E.g. Instant Messaging, or e-mail.
2. A malicious insider sends - or signals - information to a shared resource which is accessible or observable by an outsider.
   E.g. a wrongly configured public FTP, or a shared hard drive accessible from the Internet.

This means that either information is send directly to the outsider or via some mechanism that is reachable or observable for the outsider.

Based on the above two observations an important step towards data leakage prevention seems to prevent the following situations from occurring;

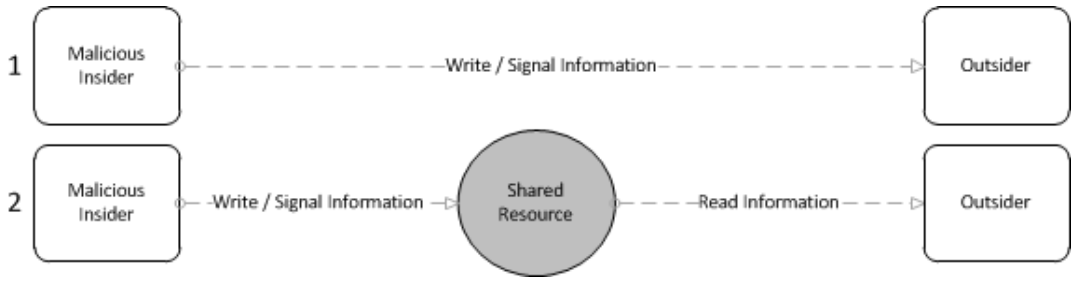


Figure 6.1: Unwanted situations, involving actions initiated by malicious insiders.

1. Preventing / Detecting malicious insiders from obtaining sensitive information;
2. Preventing / Detecting malicious insiders from sending sensitive information to outsiders;
3. Preventing / Detecting malicious insiders from sending sensitive information to shared resources;
4. Preventing / Detecting outsiders from receiving sensitive information from shared resources.
5. Preventing / Detecting outsiders from receiving sensitive information from malicious insiders;

Reducing data leakage through Option 1 can be done by controlling who has access to certain pieces of information. Most commonly this is done by only allowing information access to users of (at least) a certain `clearance', with a certain role, or which are part of a certain group. A problem with this set up is that it alone does not always protect against the `insider threat'; an insider can have legitimate clearance to access the sensitive information.

The techniques of limiting access to certain pieces of information can be also applied to Option 4 by blocking access to shared resources to unauthorized users outside of the organization, and by blocking the storage of sensitive information on shared resources for Option 3.

Not only should access to sensitive information on (shared) storage locations be limited but also the sharing of highly classified information from an insider with appropriate clearance to others without the necessary clearance (insiders, or outsiders as in Option 5). Sharing between insiders with different clearance is the problem targeted in the Bell-LaPadula model. Some difficulties in this model are that implementing the restrictions might be too hard, too costly, and / or too restrictive within a corporate setting. Applying sharing constraints only to highly sensitive information will reduce some of these restrictions.

Preventing or detecting the sending of information by insiders (Option 2) and the receiving of this information by outsiders (Option 5) directly would involve the prevention or detection of the channel used for the transfer, the combination of a particular insider and outsider, the combination of a channel used by a particular insider/outsider, the combination of insider/outsider and particular piece of information, or the combination of insider/outsider, channel and particular piece of information.

DLP Model Overview
Limiting information leakage includes limiting the possible data leakage points; this means limiting the connections and shared resources between insiders and outsiders. Removing all channels and resources is hardly ever an option. A situation in which no connections are made to the outside world creates an unworkable situation. As the first rule of Jericho mentions `The scope and level of protection should be specific and appropriate to the asset at risk' (Appendix A).

Because certain channels and shared resources cannot always be removed completely, ways to limit data leakage should be considered. Some options are applying mandatory outsider- and insider authentication, using (only) secure channels when sending sensitive information to prevent snooping, securing the information itself (encrypting, password protections), securing the insider/outsider, limiting access- and sharing options, and warning users when they attempt to send sensitive information or perform actions which might lead to information leakage.

In the model the scenarios as mentioned in the Overview are considered, as well as the 'sharing' of information between insiders within an organization.

The DLP model enforces the following:

- The outsider is authenticated;
- The insider is authenticated;
- The authentication procedure is logged;
- The outsider is sufficiently secured (see below);
- The insider is sufficiently secured (see below);
- The sensitive information is sufficiently secured during sending.
- Information on shared resources is encrypted and protected from unauthorized outsiders.
- The sensitive information retrieval to an authorized insider is logged;
- The sensitive information sending to an authorized outsider is logged.
- A notification is given to the insider prior to sending sensitive information to double-check if the insider is aware of what he is doing.

These requirements raise some questions themselves such as 'When is an outsider / insider sufficiently secured?' and 'How is information secured at all times?'. We say that something is 'secured sufficiently' when the cost of penetrating the security is deemed higher than the profit that is gained. A note to make here is that relying on the ignorance of the attacker regarding the security methods is not a proper way to secure information; a widely accepted statement is that the way your system works ('how your information is secured') will eventually be known to the attacker (Shannon's maxim). In Computer Security this is applied by using techniques which do not rely on 'security by obscurity' but instead are open and tested methods from which the security relies only on the strength of a secret key (Kerckhoffs' principle). If our model when implemented we advice to use open and tested methods and to secure points sufficiently where needed.

Cover Channel DLP Model Overview
The above DLP model offers a way to prevent data leakage through regular channels but does not handle covert channels very well. To prevent the case of a malicious insider wanting to leak information to an outsider using covert channels there is a need to look at the above data leakage model and answer the following additional questions;

- Are there any shared resources between an insider and an outsider that might be used as a covert channel?
- What channels can be used to communicate (directly) between insiders and outsiders?
- How can non-sensitive information be communicated to outsiders without creating covert channels?
- What channels can be used to communicate with each shared resource?
- What channels can be used to communicate between shared resources?
- What information should not be allowed to be send to outsiders?
- If there is a need to send sensitive information, (how) can this be done?

Our covert channel DLP model incorporates the above ideas. This model tries to protect information by making sure that users (insiders and outsiders) are authenticated before they can access or transfer sensitive information, that the information they request and send is not sensitive information, and that the channels used are not used to transfer hidden information.

In our Covert Channel model we have made some changes to the DLP model:

- All channels must be secured;
- To prevent signalling to outsiders the model includes Channel Analysis.

By applying Channel Analysis techniques hidden sensitive information within the channel itself can be found. Secure channels remove the possibility of channel snooping.

6.2 Data Leakage Prevention Model.

Background

DLP is currently a field covered by several vendors such as Symantec (Symantec Data Loss Prevention), Code Green Networks (TrueDLP), McAfee (McAfee Data Loss Prevention), CA Technologies (CA DLP), and IBM (Fidelis). Most of these vendors originated from other fields such as the anti-virus-, anti-malware-, or firewall-business, the network behaviour analysis market, or compliance/financial-, or IDS/IPS industry. Through expansion of their own products and/or through takeovers of small specialist companies these companies have entered the DLP market. As mentioned in chapter 5.4 there are different classical approaches to DLP; Solutions range from products that filter out known sensitive information from network traffic, while others focus on preventing the information from leaving an insider's system before it enters the network.

The main difference between these projects, according to Rustem Khayretdinov (Vice President of Infowatch.com) is the channels they can control (E-mail, data copying on USB stick, Skype, etcetera) and the ways of analysing data [1]

To determine where to secure information all three data categories (data in motion, at rest, and in use) need to be considered to minimize the chance of information leakage. Solely protecting data on devices is not enough because it is difficult, impractical and sometimes even impossible to ensure that all equipment within a company network is equipped with a (host-based) DLP system that monitors information leaving the system (e.g., visitor- and customer laptops accessing the network).

To leak information, it does not need to be 'in use' at an endpoint: copying information from a server or storage location to an outsider's server is another way information can leak.

When focusing on data in motion, models as displayed in the 'Information Flow models' chapter can be used because they model where (and what type of) information is allowed to (theoretically) flow between different insiders. Adding data leakage prevention to these models would mean determining where and how to apply the different techniques in such a model.

The separation of DLP systems in Network-based and Host-based approaches is relevant, but the two have to complement each other when trying to tackle the data leakage problem as a whole; It would be a missed opportunity to create a model which focuses solely on host-based endpoint security while disregarding obvious possible leakage points such as equipment for which host-based DLP systems are not practical or possible, or vice versa. In the model this should be reflected by modelling one comprehensive approach which does not centralize on one of the two approaches, but which rather takes the organization's sensitive information as a focal point and applies protection techniques where appropriate.

The model takes the following points as its core way of defeating information leakage; the protection of the information itself, the security of all points on which information resides, and the security of the communications used to transfer sensitive information.

In chapter 5 Figure 5.3 shows a network with unwanted connections to the outside world. In this situation, the network is protected using the classical approach of a corporate firewall. The approach we use in our DLP model is based on the Jericho principle of de-perimeterization, the Bell-LaPadula idea of classifications and clearances, and the Chinese Wall idea of protection against conflicts of interest. This should ensure that only people with need-to-know and sufficient clearance can access the information, and when the information is used that it stays secure and does not flow to people without the appropriate rights or who might have a conflict of interest.

Securing the information can be done at different levels; securing the file itself, the server it is stored on, the insider or outsider it is transferred to, while in use, while it leaves the corporate network, the communication channels between insiders, outsiders, or shared resources. Figures 6.2 and 6.3 show this idea.

Completely locking down a machine is the ideal situation; Perfect encryption, endpoint security, and no incoming / outgoing connections reduce the possibility of data leakage to practically zero. However, this also reduces the usability to zero. At some point, the information needs to be decrypted and will be viewed by someone. Once this

[1] http://wn.com/Data_Leak_Prevention

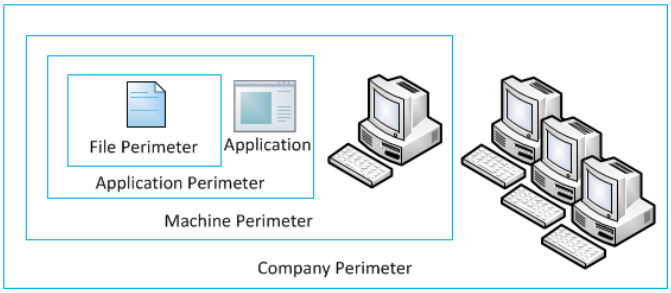


Figure 6.2: Protection perimeters - Abstract representation.

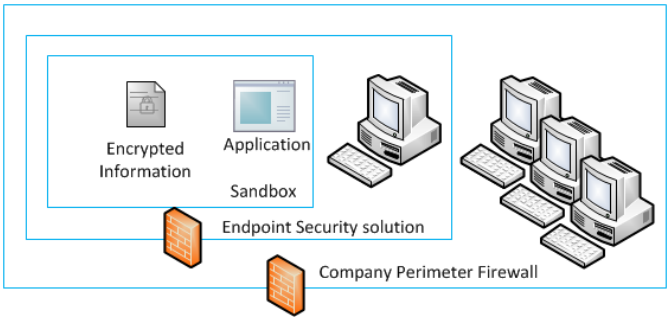


Figure 6.3: Protection perimeters - Example implementation.

has occurred there is already the possibility of data leakage (e.g. by memorizing the information and writing it down later). Technical flaws (such as imperfect encryption) are other factors that might allow data leakage. No matter how secure the DLP solution, there will always be a possibility of data leakage.

In our model, when encryption is used we assume it is unbreakable, when a perimeter is established we deem it impregnable. In reality this is of course not the case.

Information is exchanged frequently between different parties. Almost all of the time only one of the parties can be controlled directly ('your company') while the other party might have a totally different IT infrastructure and security level. This means that;

- The connection might be insecure;
- The recipient is unauthorized / unprotected.

To remove these dangers from the model we can require that all connections used to send sensitive information are secure connections. In Jericho terms this means using only open, secure protocols to communicate. Although the other party cannot be adjusted directly you can enforce the usage of for example secure connections (e.g., communicate only using SSLv3, else communication ends).

The danger on the side of a leaking recipient could be reduced by only allowing sensitive information to be sent to authorized recipients who have authenticated themselves and have appropriate clearance for the sensitive information they wish to receive. We include this in our model.

Model

Our model for DLP detection and prevention consists of these possible communications:

insider - Outsider Sending sensitive information from an insider to an outsider directly.

insider - Outsider Sending non-sensitive information from an insider to an outsider directly.

Shared Resource - Outsider An outsider can access sensitive information on a shared resource.

Shared Resource - insider an insider can access sensitive information on a shared resource.

insider - insider an insider sending sensitive information directly to another insider.

Analogous to the above four situations that deal with sensitive information the same situations occur with non-sensitive information. Only the situation where non-sensitive information is sent directly from an insider to an outsider has been depicted in the model.

The next few sections go into more detail regarding each of the situations for data leakage prevention of sensitive information as well one situation which deals with non-sensitive information (insider to outsider directly).

Each situation is shown in a separate subsection below.

Sensitive information: Insider - Outsider

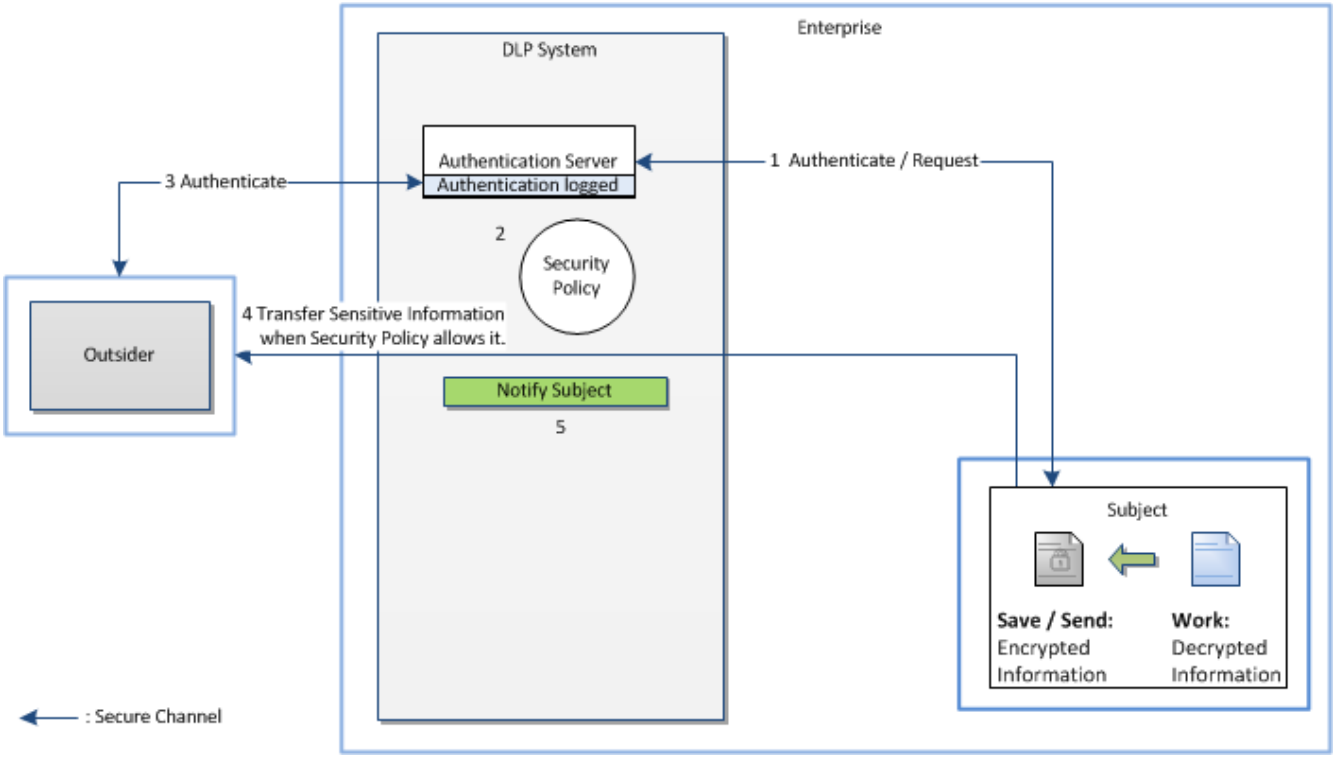


Figure 6.4: DLP Model - Sending sensitive information from insider to Outsider.

The process of sending sensitive information to an outsider is depicted in the above model (the numbers in the model reflect the steps in the following list).

1. To send sensitive information from an insider to an outsider the insider authenticates himself and sends the request to the DLP system that he wants to send sensitive information to a particular outsider.
2. The DLP system checks if the authentication succeeds and if the insider is allowed to send this particular sensitive information to the outsider as per the Security policy. The Security Policy should contain information on what information this insider is allowed to share and with whom. If the insider is not authenticated, or does not have the authorization to send information to this particular outsider, the communication ends.
3. After insider authentication and information sending request the outsider authenticates himself.
4. If the authentication check is passed the information is send to the outsider (through the DLP system), else it is not.
5. If the insider tried to send sensitive information that he was not allowed to send he is notified that he (attempted to) send sensitive information.

The information to be send is encrypted during saving on the insider's hard drive as well as during transfer. Only when the insider is actively working with the information it is decrypted.

The Jericho principle is applied in this model; the 'rectangles' surrounding the insider and the outsider indicate that they are secured (Host-based security), the rectangle surrounding the DLP system and the insider indicate the Enterprise is secured (e.g., corporate firewall), and the information itself is secure by applying encryption.

Sensitive information: Shared Resource - Outsider

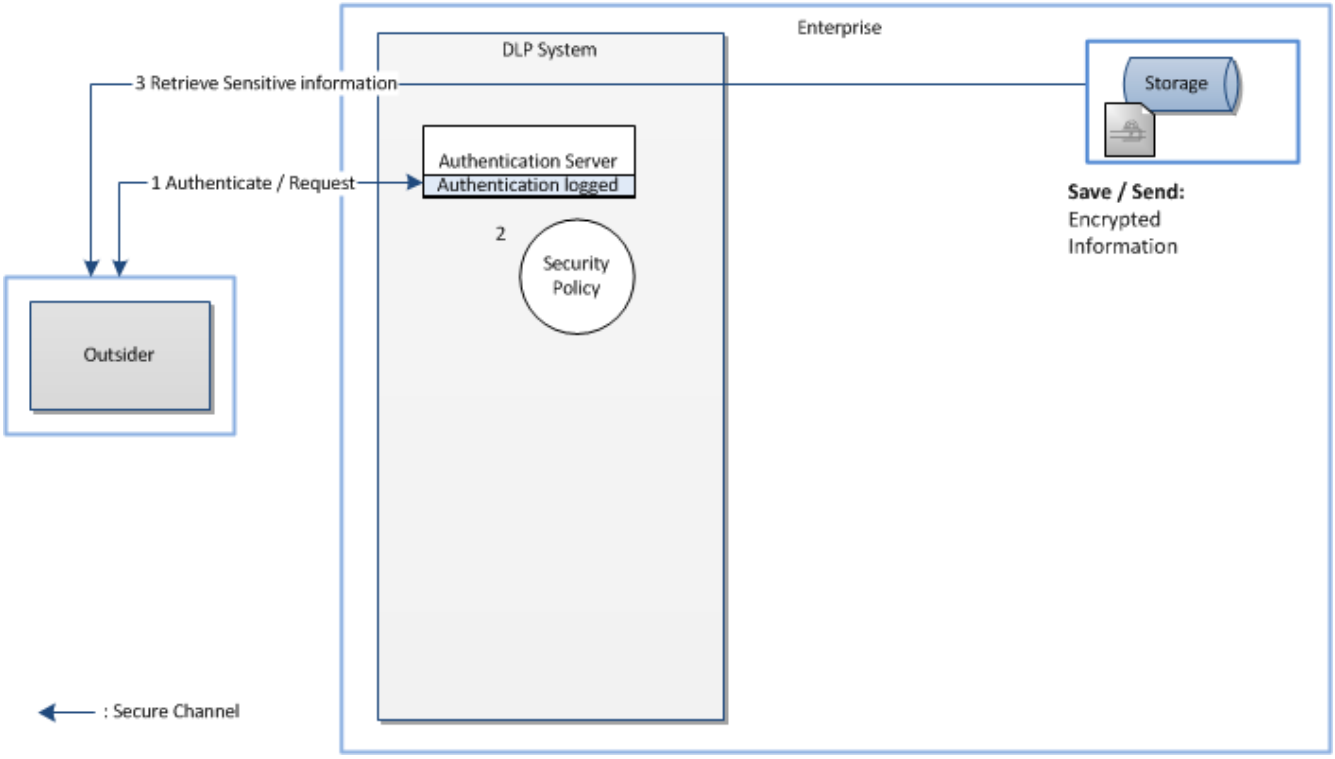


Figure 6.5: DLP Model - Outsider retrieving sensitive information from a shared resource.

An outsider accessing sensitive information stored on a shared resource is the DLP problem this model focuses on. In this situation there is no insider within the enterprise directly involved in the transfer of information. Of course, an insider within the enterprise might have put the information here for the outsider to retrieve (this is not included in the model). The information on the shared storage resource is encrypted. The 'square' surrounding the outsider indicates that he is considered 'secure' in the model.

- The outsider sends a request to the DLP system with the notice that he wants to receive the particular piece of sensitive information from the shared resource together with his authentication information.
- The DLP system checks if the Security Policy allows the outsider to retrieve the requested information from the shared resource. The Security Policy should contain information on what information the shared resource is allowed to share and with whom. If the outsider is not authenticated, or does not have the authorization to receive information from this resource, the communication ends.
- If the check is passed the outsider can access the shared resource and access (only) the requested particular piece of information.

Sensitive information: Shared Resource - insider

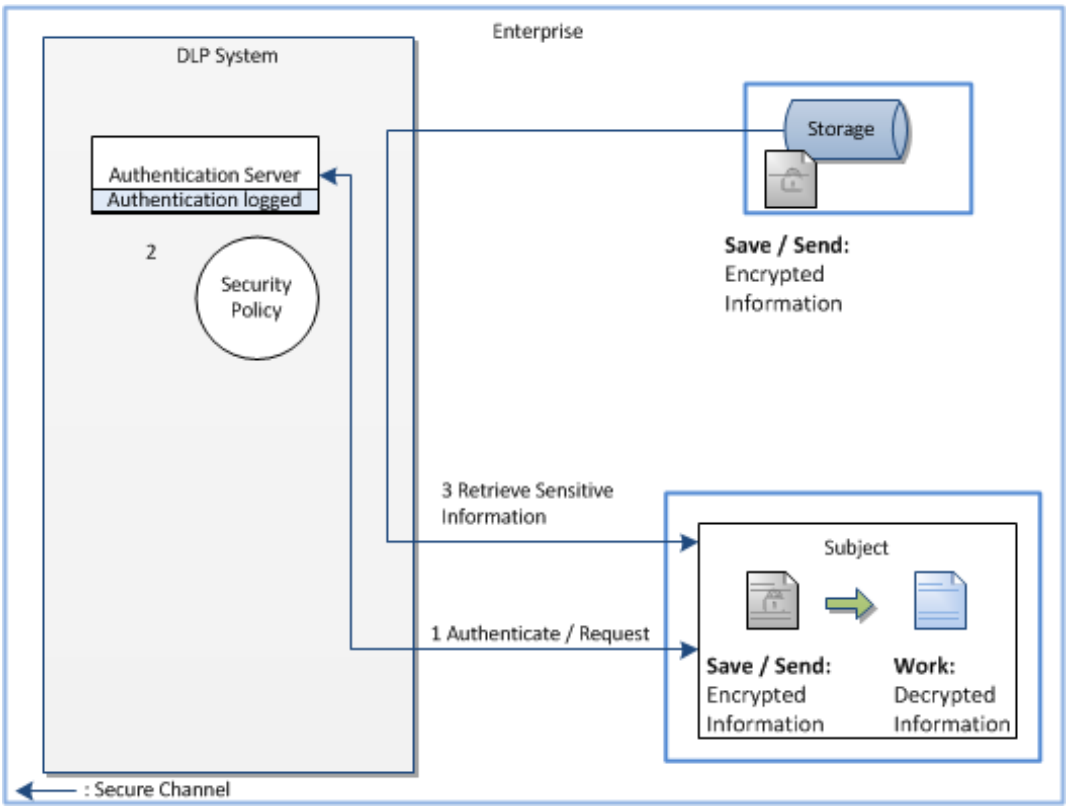


Figure 6.6: DLP Model - insider retrieving sensitive information from a shared resource.

Accessing sensitive information from a shared resource to an insider is performed in three separate steps:

1. To retrieve sensitive information from a (shared) storage location the insider authenticates himself and sends the request to the DLP system that he wants to receive sensitive information from a particular resource.
2. The DLP system checks if the authentication succeeds and if the insider is allowed to receive sensitive information from this resource as per the Security policy. The Security Policy should contain information on what information the shared resource is allowed to share and with whom. If the insider is not authenticated, or does not have the authorization to receive information from this resource, the communication ends.
3. If the authentication check is passed the information is send to the insider (through the DLP system), else it is not.

Information retrieval in the DLP Model for Shared Resources does not differ much for Outsiders or Insiders (insiders). In many companies the internal network is more loosely secured than connections to the outside. The reason for having internal traffic flow through the DLP System as well is to limit the number of copies of certain information only to people that have a 'need-to-know' for the requested sensitive information.

In practice, connections to the outside might be secured more strongly. An implementation of this model might employ Access Control Lists only, while for communications from a shared resource to an outsider Access Control Lists, IP based filtering/Whitelisting, and additional Firewall/Inspection checks are performed. The model, however, abstracts from this difference.

Sensitive information: insider - insider

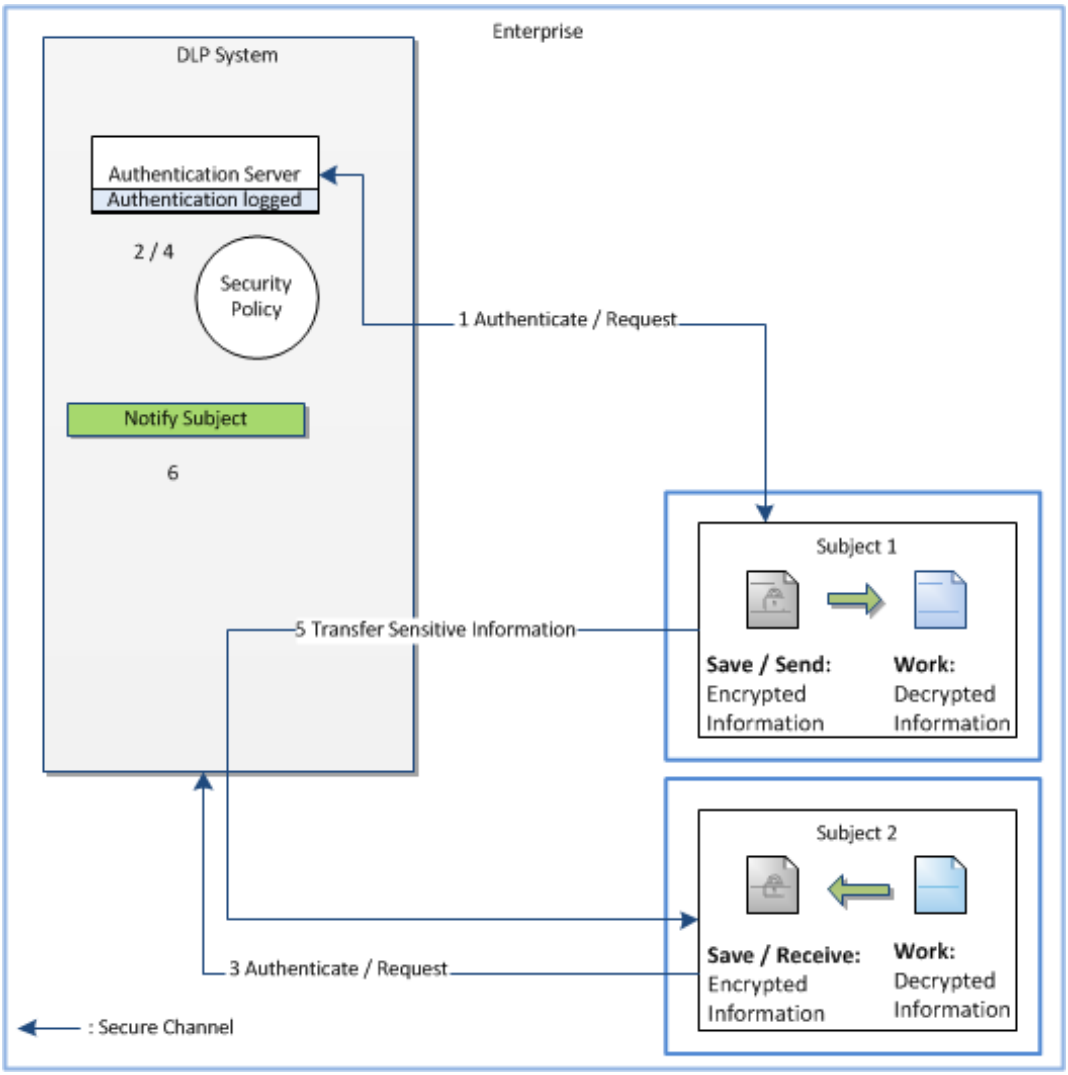


Figure 6.7: DLP Model - insider sharing sensitive information with another insider.

Sharing sensitive information from one insider to another insider is performed in these steps:

1. The first insider authenticates himself and sends a request to the DLP system that he wants to send a particular piece of sensitive information to another insider;
2. The DLP system checks if the authentication of the first insider succeeds and if the insider is allowed to send this sensitive information to the other insider per the Security policy. The Security Policy should contain information on what information the first insider is allowed to send and with whom. This includes information on conflict of interest. If the insider is not authenticated the communication ends. If the insider does not have the authorization to send this information, he is notified that he attempted to send sensitive information, and the communication ends.
3. The second insider authenticates himself and sends a request to the DLP system that he wants to receive a particular piece of sensitive information from another insider;
4. The DLP system checks if the authentication of the second insider succeeds and if the insider is allowed to receive this sensitive information from the other insider per the Security policy. The Security Policy should contain information on what information the second insider is allowed to receive and from whom. If the insider is not authenticated, or does not have the authorization to receive this information, the communication ends.
5. If the check is passed the information is sent from the first insider to the second insider through the DLP system, else it is not.

6. If the first insider tried to send sensitive information that he was not allowed to send he is notified that he (attempted to) send sensitive information.

Sharing information between insiders is included in the DLP model for two reasons:

- Internal Conflict of Interest (refer to the Chinese Wall model): The sensitive information accessible to one insider might be of interest to another insider (within the same company) who is not allowed to receive this information. This other insider might abuse this information for his own benefit.
- Different rights of insiders regarding sensitive information (refer to the Bell-LaPadula model): When one insider within an organization can only receive sensitive information and share it with other insiders, and another insider is able to receive information from insiders and send it to outsiders this can cause information leakage when the two insiders cooperate together.

Non-Sensitive information: insider - Outsider

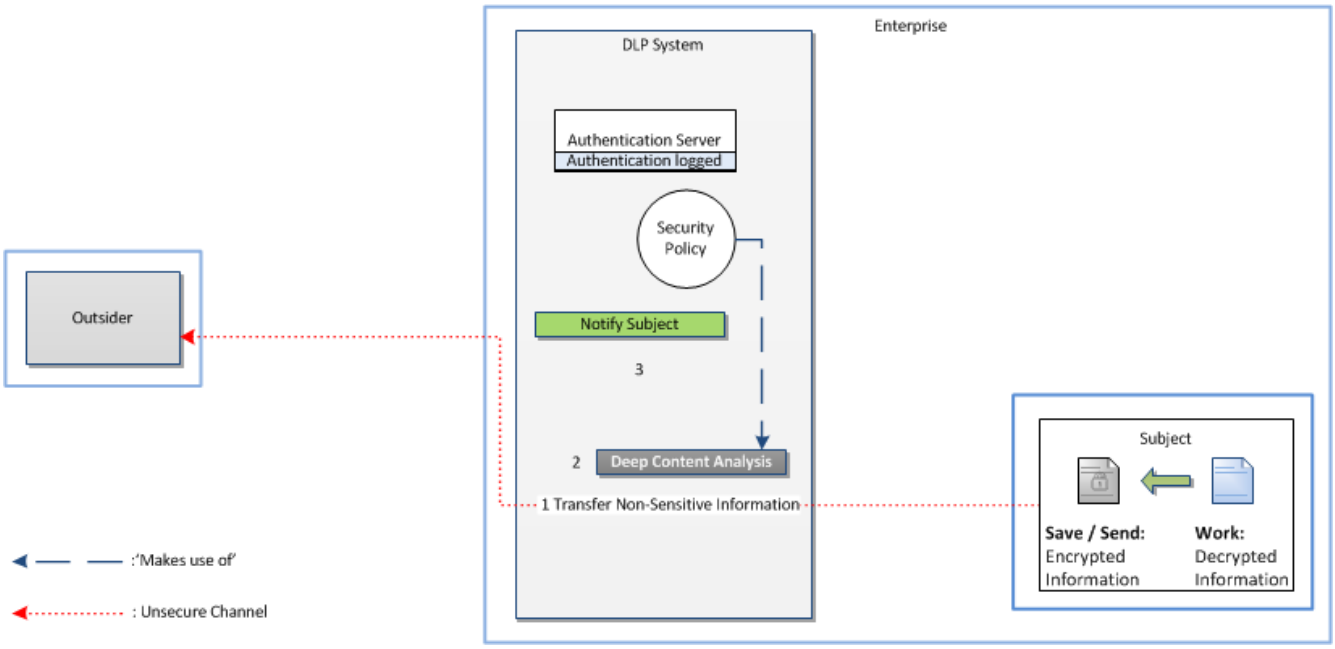


Figure 6.8: DLP Model - insider sending non-sensitive information to an Outsider.

Sending non-sensitive information from an insider to an outsider is performed in three separate steps:

1. The insider sends the information through the DLP system to the outsider. As this transfer involves no sensitive information the use of authentication and secure channel is not enforced.
2. The DLP system performs Deep Content Analysis on the non-sensitive information to detect if it does not (secretly) contain sensitive information. The Security Policy should contain information on what information the insider is allowed to send and to whom.
3. If sensitive information is found the transfer is blocked, otherwise it is passed to the outsider. If the information is sensitive the insider receives a notification.

The Deep Content Analysis (DCA) technique uses the Security Policy to create the distinction between sensitive and non-sensitive information. Refer to section 5.6 for possible DCA methods.

Situations

In this section we will reflect on some real-life situations regarding data leakage and investigate if and how the model represents the data leakage event:

Event	Protection
Printing sensitive files.	Host-based DLP will monitor the print request from the insider and detect that the file to be printed is a sensitive file. Depending on certain factors (such as the identity of the person who is printing) it can block the print request or send an alert message. Depending on the sensitivity of the document this can be considered an instance of sending information from an insider to an Outsider, or from one insider to another (the printer being the outsider in the first, or another insider in the latter).
Copying sensitive files to USB-stick.	Host-based DLP will monitor the copy request from the insider (Operating System) to the outsider (USB-stick) and detect that the file to be saved is a sensitive file. Depending on the security policy this is allowed or blocked.
Copying part of a sensitive document to an unclassified document and send it as 'non-sensitive' information.	The unclassified message can be caught by Network-Based DLP systems that use Deep Content Analysis (DCA) techniques. DCA can detect partly copied information using partial document matching techniques and request the DLP system to block the transfer.
E-mailing top secret blueprints to a competitor.	The DLP system will not allow the competitor to receive information when sent through the regular sensitive information procedure; The competitor will not be in the `authorized receivers' list and the blueprint is sensitive information which the DLP does not allow to be send.
Uploading files to an online storage website (e.g., dropbox)	The online storage website will be considered an Outsider. If the file is non-sensitive it will be monitored (and blocked if sensitive information is detected) by the DLP system. When the file is sensitive information an encrypted channel will be used to transfer the file (if the uploader is allowed to transfer this sensitive information to the online storage website).
Photographing a document	Once an insider has access to a sensitive document it is not possible for a DLP system to prevent him from making a photograph/copy from the document by means of hardware that is in no way connected to the network (e.g., a camera, or a private GSM phone). Other measures to prevent this should be taken (such as training, cameras in the building, a company policy forbidding cameras/GSMs in the building, ..).

Table 6.1: Data Leakage situations and their relation to the DLP models.

6.3 Covert Channel DLP Model.

The Covert Channel DLP model is based on the DLP models with these distinct changes; all channels are secured and channel analysis is performed on all channels to reduce covert channel existence.

Channels secured
All channels are secured in the CC model to prevent information from leaking by outsiders snooping the channel or intercepting sensitive information. We no longer allow the sending of any (including non-sensitive) information via unsecured channels. All information, both sensitive and non-sensitive, has to be sent via secure channels.

Channel Analysis
Because manipulation of the channel is still possible (and can be used to signal a message to an outsider) Channel Analysis is performed on transfers through secure channels. Channel Analysis (CA) is the practice of inspecting the channel for oddities which might suggest that an insider is sending a message through a covert channel. The difference with Deep Content Analysis is that not the information being sent is inspected but the channel itself.

Usable techniques include;

- Protocol Anomaly Detection (e.g., irregular data in protocol fields);
- Network Behavior Anomaly Detection (e.g., TCP/UDP fanout, IP fanout, IP/MAC spoofing);
- Trend Inspection (e.g., certain sequences of traffic at predictable times that should not occur) ;
- Traffic Volume / Bandwidth Use Monitoring (e.g., sudden and unexpected increase in particular traffic)

The CC DLP model tries to reduce a number of channels (see below 'Transfer Channels), but for the channels that remain Deep Content Analysis and Channel Analysis are techniques that carry out most of the work. Consider the following scenario:

- A malicious insider places a file containing the first 3 letters from a sensitive document he has access too;
- The insider transfers this file to an outsider. Without DCA/CA this is likely to be allowed. The outsider now has the first 3 letters of a sensitive document;
- The insider changes the file to contain the next 3 letters from the sensitive document;
- The insider transfers this file to the outsider. The outsider now has the first 6 letters of a sensitive document;
- Etcetera;

Traditional DCA techniques such as rule-based/regular expressions, database fingerprinting, exact file matching, partial document matching, statistical analysis, conceptual/lexicon analyses, and category matching (refer to 5.6) will not detect such leakage, while new DCA techniques such as profiling or machine learning techniques might (refer to 5.6).

Additionally, CA techniques might detect this as well; numerous requests from one IP to get a single small text file many times should raise suspicion.

Transfer Channels
Reducing the chance of covert channels is started by reducing the number of channels that can be abused as covert channels. Regarding the existence of transfer channels there are three situations, with respect to covert channels and shared resources:

1. Remove the possibility for an outsider to access shared resources. This way the possibility of covert storage channels is eliminated.
2. Force all information sharing with outsiders through shared resources. This way a lot of other possible covert channels can be blocked ('block all communications, except from/to the shared resource').
3. Allow both. This offers the most usability, but also poses the most risks.

When directly sending a file to an outsider, it is also transferred via the DLP system. The communication channel in the DLP model is encrypted, but still the channel is set up and controlled by the insider. When this insider is malicious he can use this power to manipulate the channel itself to send information. In our covert channel DLP model we remove the power of an insider to manipulate a channel by moving the channel creation and maintenance from the insider to the DLP system. In a way, the DLP itself becomes a 'shared resource'. On this self-made channel the DCA and CA techniques should still be applied.

To create the CC DLP models from the DLP models we thus adjust them with the above changes, and discard the DLP model that handles non-sensitive information transfer.

Each situation is shown in a separate subsection below.

Sending information: insider - Outsider

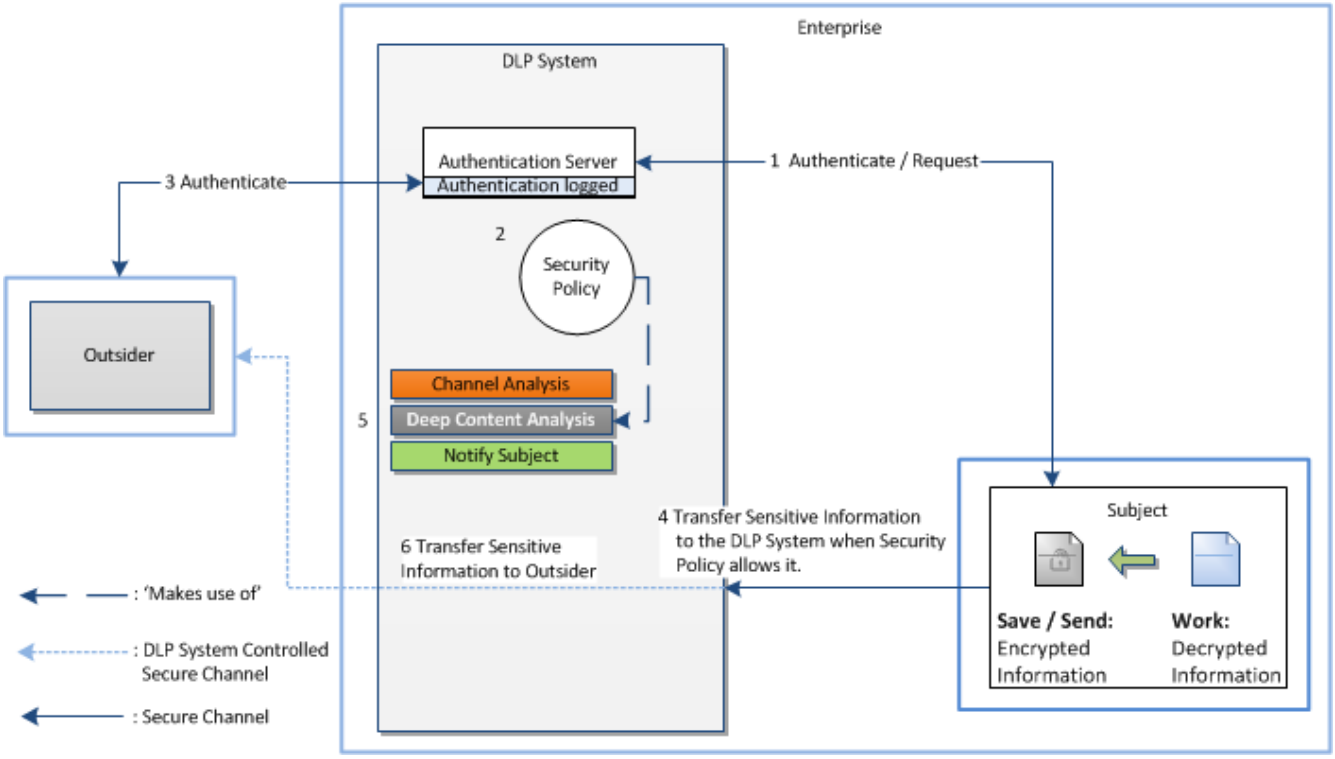


Figure 6.9: CC DLP Model - Sending sensitive information from insider to Outsider.

The process of sending sensitive information to an outsider is depicted in the above model. On all communications, Channel Analysis and Deep Content Analysis is applied. Although the direct transfer is set up by the DLP system, not all risks of covert channels are gone (e.g., the '3 letters' example).

1. To send sensitive information from an insider to an outsider the insider authenticates himself and sends the request to the DLP system that he wants to send sensitive information to a particular outsider.
2. The DLP system checks if the authentication succeeds and if the insider is allowed to send this particular sensitive information to the outsider as per the Security policy. The Security Policy should contain information on what information this insider is allowed to share and with whom. If the insider is not authenticated, or does not have the authorization to send information to this particular outsider, the communication ends.
3. After insider authentication and information sending request the outsider authenticates himself.
4. If the authentication check is passed the information is sent from the insider to the DLP system, else it is not.
5. The DLP system takes the information from the insider and transfers it to the Outsider via a channel between itself and the Outsider. The DLP system creates this channel.
6. If the insider tried to send sensitive information that he was not allowed to send he is notified that he (attempted to) send sensitive information.

The Jericho principle is applied in this model; the 'rectangles' surrounding the insider and the outsider indicate that they are secured (Host-based security), the rectangle surrounding the DLP system and the insider indicate the Enterprise is secured (e.g., corporate firewall), and the information itself is secure by applying encryption.

Sending Information: Shared Resource - Outsider

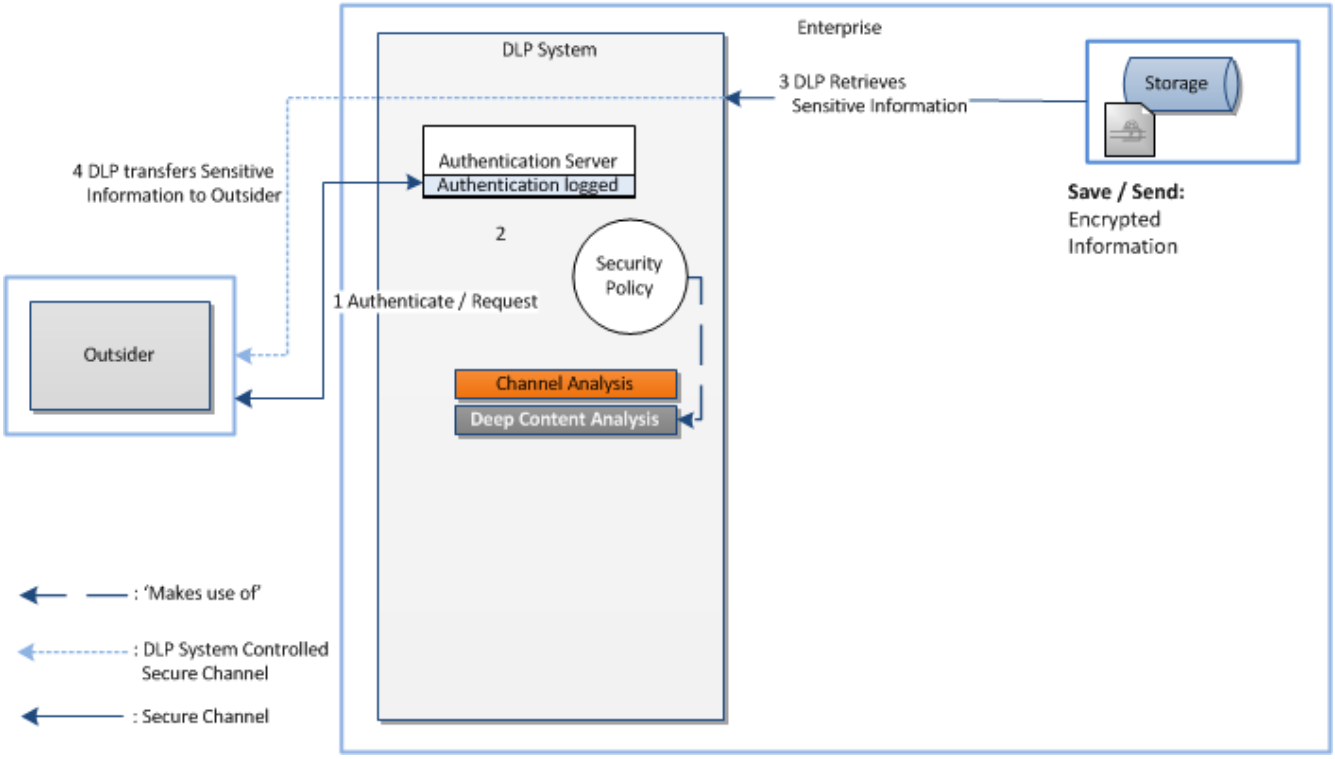


Figure 6.10: CC DLP Model - Outsider retrieving sensitive information from a shared resource.

This model applies to the situation in which an outsider requests a file from a shared (storage) device within the enterprise. In this situation there is no insider within the enterprise directly involved in the transfer of information. Of course, an insider within the enterprise might have put the information here for the outsider to retrieve. This is not directly included in the model but is incorporated by applying DCA/CA on the information transfer channel. The information on the shared storage resource is encrypted. The 'square' surrounding the outsider indicates that he is considered 'secure' in the model.

- The outsider sends a request to the DLP system with the notice that he wants to receive the particular piece of sensitive information from the shared resource together with his authentication information.
- The DLP system checks if the Security Policy allows the outsider to retrieve the requested information from the shared resource. The Security Policy should contain information on what information the shared resource is allowed to share and with whom. If the outsider is not authenticated, or does not have the authorization to receive information from this resource, the communication ends.
- If the check is passed the DLP system accesses the shared resource and transfers (only) the requested particular piece of information.
- The DLP system sends the information obtained from the shared resource to the outsider, applying DCA and channel analysis before and during the transfer.

Sending information: Shared Resource - insider

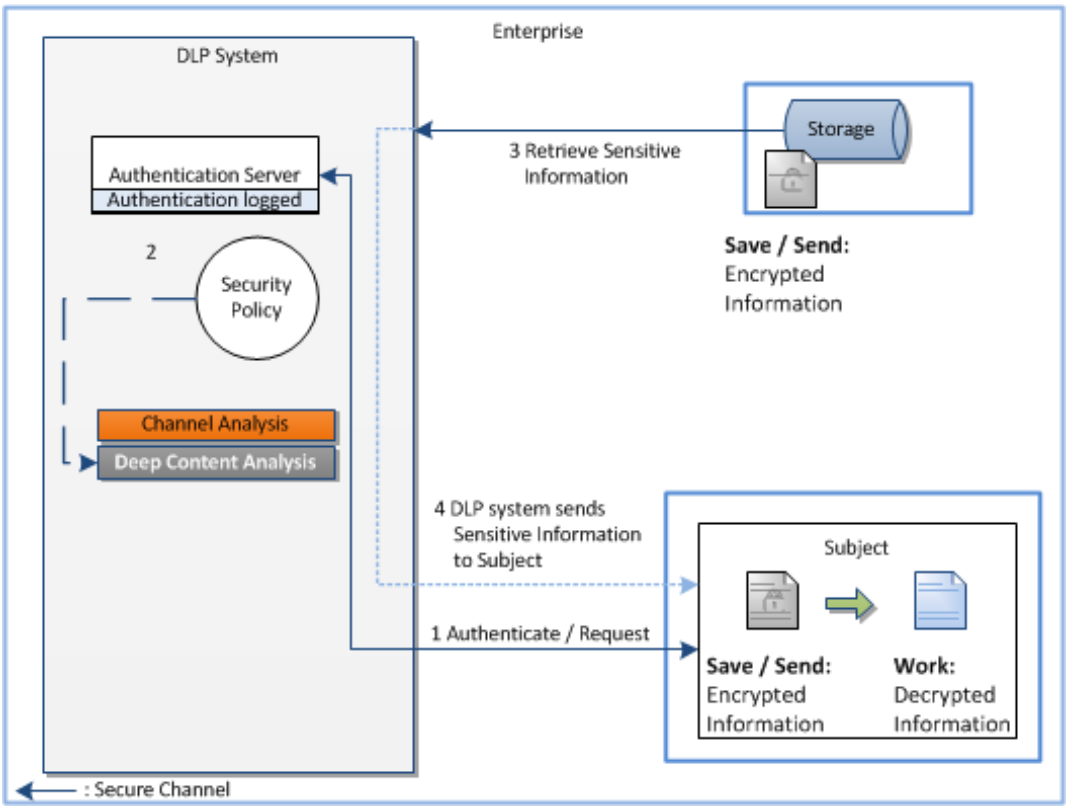


Figure 6.11: CC DLP Model - insider retrieving sensitive information from a shared resource.

Information transfer from a shared resource (storage) location to an insider also goes through the DLP system:

1. To retrieve sensitive information from a (shared) storage location the insider authenticates himself and sends the request to the DLP system that he wants to receive sensitive information from a particular resource.
2. The DLP system checks if the authentication succeeds and if the insider is allowed to receive sensitive information from this resource as per the Security policy. The Security Policy should contain information on what information the shared resource is allowed to share and with whom. If the insider is not authenticated, or does not have the authorization to receive information from this resource, the communication ends.
3. If the check is passed the DLP system accesses the shared resource and transfers (only) the requested particular piece of information.
4. The DLP system sends the information obtained from the shared resource to the insider, applying DCA and channel analysis before and during the transfer.

Analogous to the DLP Model for Shared Resources the function of having internal traffic flow through the DLP System is to limit the number of copies of certain information only to people that have a 'need-to-know' for the requested sensitive information.

Sending information: insider - insider

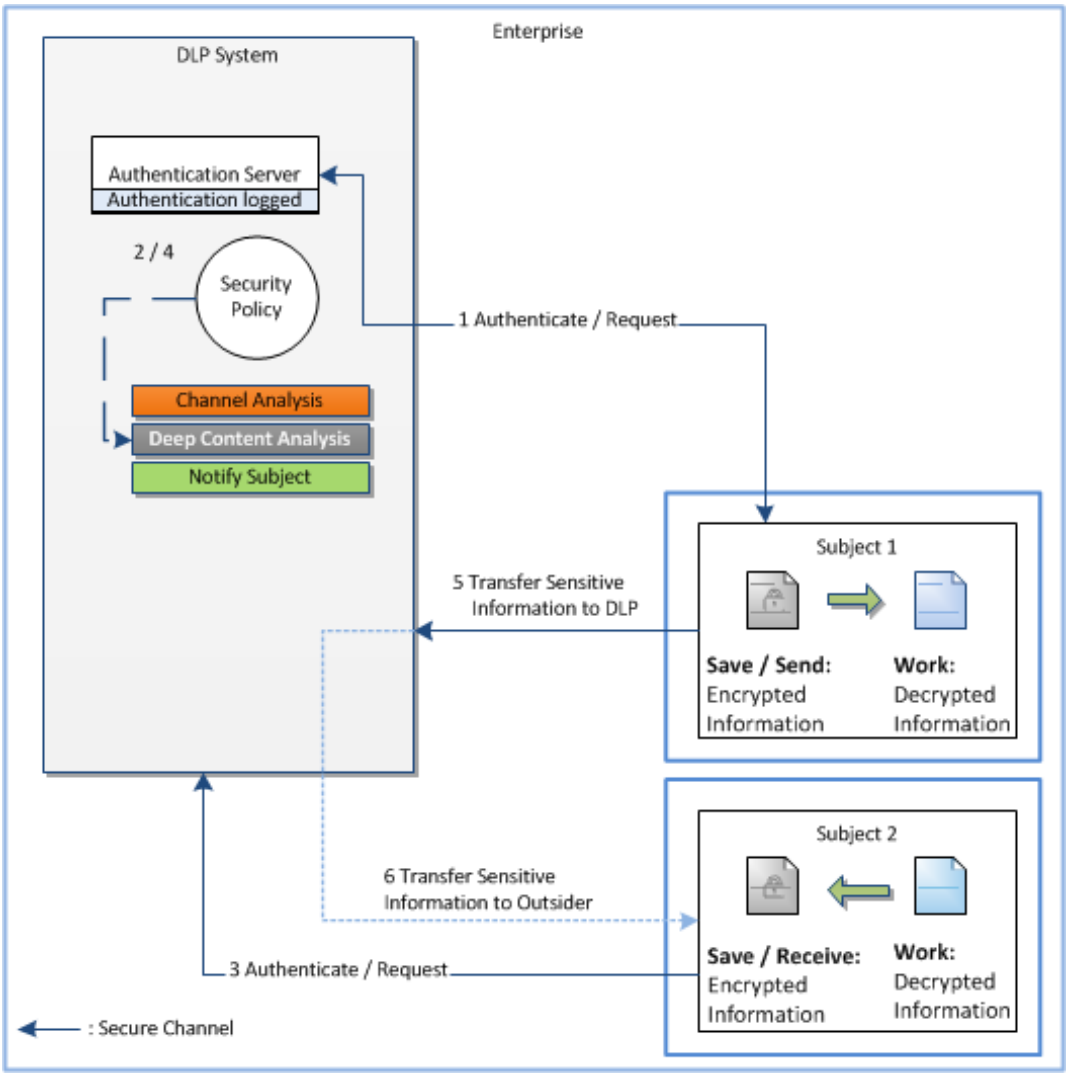


Figure 6.12: CC DLP Model - insider sharing information with another insider.

The process of sharing information between two insiders is depicted in the above model.

1. The first insider authenticates himself and sends a request to the DLP system that he wants to send a particular piece of information to another insider;
2. The DLP system checks if the authentication of the first insider succeeds and if the insider is allowed to send this information to the other insider per the Security policy. The Security Policy should contain information on what information the first insider is allowed to send and with whom. This includes information on conflict of interest. If the insider is not authenticated the communication ends. If the insider does not have the authorization to send this information, he is notified that he attempted to send sensitive information, and the communication ends.
3. The second insider authenticates himself and sends a request to the DLP system that he wants to receive a particular piece of information from another insider;
4. The DLP system checks if the authentication of the second insider succeeds and if the insider is allowed to receive this information from the other insider per the Security policy. The Security Policy should contain information on what information the second insider is allowed to receive and from whom. If the insider is not authenticated, or does not have the authorization to receive this information, the communication ends.
5. If the check is passed the information is send from the first insider to the DLP system, else it is not.

6. The DLP system sends the information obtained from the first insider to the second insider, applying DCA and channel analysis before and during the transfer.

Sharing information between insiders is included in the CC DLP model for the same reasons why it is included in the DLP model;

- Internal Conflict of Interest (refer to the Chinese Wall model): The sensitive information accessible to one insider might be of interest to another insider (within the same company) who is not allowed to receive this information. This other insider might abuse this information for his own benefit.
- Different rights of insiders regarding sensitive information (refer to the Bell-LaPadula model): When one insider within an organization can only receive sensitive information and share it with other employees, and another insider is able to receive information from insiders and send it to outsiders this can cause information leakage when the two insiders cooperate together.

The appliance of DCA/CA techniques on the transfer channel should reduce the possibility of one insider sending sensitive information to another insider who is not authorized to receive this information by manipulating the channel when sending a non-sensitive file (the CC DLP models treat all information as 'sensitive').

Chapter 7

Security Policy

The Security Policy and supporting processes define the practices for identifying, containing, responding and mitigating incidents. In this chapter we will focus on the DLP part of a Security policy. We will start on a high-level and look into the required governance structure, the classification and monitoring process, and the response handling. In a separate section we research the more implementation-focused 'security policy' as referenced in earlier chapters.

***

7.1 Governance

To support the DLP process a governance structure needs to be in place to provide structure regarding responsibilities. Although different for each company there are some common players that can typically be found in a global organization:

- Security Incident Response Team:
  - Information Security Manager;
  - Risk Managers.
- IT Operations;
- Legal;
- HR.

The Security Incident Response Team (SIRT) is accountable for the implementation and maintenance of the DLP strategy, policy, and supporting policy. The Information Security Manager is part of this team together with other (regional) Risk Managers.

The Information Security Manager is the global lead for Information Security & Incident Management while the Risk Managers are commonly the regional lead for Information Security & Incident Management. The Risk Managers are responsible for determining the required resources that are needed for a particular incident within the region he is assigned to.

IT operations are responsible to support the Risk Manager where necessary. The legal department is responsible for advising the Security Incident Response Team where legal advice is required. Human Resources is responsible for handling personnel issues that may arise during and subsequent to an event.

7.2 Classification and Monitoring

One of the processes typically involved in DLP is classification. When a document or piece of information is assigned a certain information classification level (such as Public information, Confidential, Sensitive, etc.) this creates specific responsibilities regarding storage, handling, deletion, sharing and transfer of this information for information owners and users. Exactly what these responsibilities are should be described in the company End User policy. Typically, the information owner is responsible for assigning a classification to the information and for providing oversight of proper information handling in accordance to the given classification. The Information Security Manager is typically responsible for changes to the data classification policy.

An important aspect of the Security Policy is defining what counts as an incident and how these incidents can be reported. In a DLP solution what typically counts as an incident are security incidents where there is knowledge (or reasonable belief) that unauthorized disclosure of sensitive information has taken place. This information can include (combinations of) Personally Identifiable Information (PII), secret/confidential documentation, private communications, and more.

Indications and precursors of incidents can be reported via different channels:

- User Notification: Notifications reported by employees or contractors who observed a DLP incident;
- External Notifications: This includes notifications reported by third parties, government agencies (law enforcement, intelligence agencies), and public sources.
- System Notification: Monitoring solutions within different tools can typically generate alerts. This includes software such as IDS, IPS, anti-virus software, anti-malware software, file integrity checking software, security information and event management tools (SIEM), and monitoring solutions within the DLP software.

The Information Security & Incident Management team is notified of these incidents. Monitoring within DLP software starts by detection of the (precursor of an) incident.

The policy should contain or refer to the information that is considered sensitive. The first step in this process is determining what information within the organization should be considered sensitive information and thus should be protected (classification).

Appropriate classification should be based on several factors including;

- Legal- and regulatory requirements;
- Sensitivity and criticality;
- Impact of loss or leakage;
- Risks and threats to the information.

Legal- and regulatory requirements includes legislation regarding the secure handling of Personally Identifiable Information (PII) and credit card information. The fines for not handling this data appropriately are significant. Upcoming plans for legislation regarding data loss incidents mandates organisations to inform the Dutch data protection authority (CBP) and affected clients ('Meldplicht Datalekken').

There are different ways information classification handled or ways this process can be assisted:

- Creating filters; Creating filters (such as key words, expressions, etcetera) that will create a classification based on (combinations of) words within the information. In Chapter 5.6 (traditional / new DCA techniques) different content inspection techniques were introduced. These techniques can be used (to assist) in determining the classification of information.
- Manual classification the information files; The information owner classifies the information file. As the owner of the file is aware of the data handling
- No classification. No classification is done beforehand. Sensitive data is detected by applying DCA techniques when the information is in transfer.

A DLP solution monitoring detects information leakage through the use of the classification level and the contents of the information itself (content), but can also use contextual information such as the username or accountname of the sender / receiver, the IP information, geographical location, etcetera. Both these types of information can be useful in detecting information leakage.

7.3 Response

Security events can be gathered in different categories. For DLP incidents these categories are similar to the classifications used for general security incidents:

- Critical: DLP incident affecting business critical production systems, highly confidential information, sensitive assets, and/or has a significant effect on the productivity of a large number of users.
- Major: Affecting non-critical production systems, non-sensitive systems, confidential information, and/or has an effect on multiple a significant number users.
- Minor: Only affects preproduction systems, workstations, and/or only affects a limited number of local users.

Depending on the criticality of the incident the response time to take action is influenced. As a guideline, for critical incidents this is in the range of within 30 minutes, for major incidents within 2 hours, and for minor incidents between 4 to 8 hours.

A response starts by gathering information regarding the incident and making a first estimate regarding severity and priority. The next step is damage control to limit the impact of the incident. Steps necessary to contain the incident are initiated followed by operations to stop and solve the incident. A more comprehensive collection of relevant facts and information is gathered and a deeper analysis of the incident is performed. If applicable, systems that were unable to function properly are returned to normal business operations. The incident is documented and the relevant internal and external parties are informed.

After an incident has been solved the 'lessons learned' are to be evaluated and incorporated in the DLP strategy and policies to reduce the occurrence and impact of similar incidents in the future.

7.4 Implementation

In earlier chapters there were references to the 'security policy' as well. These references were not always pointed to the high-level security documentation typically found in larger companies which was described earlier in this chapter but to the more low level implementation of a DLP policy which contains all the information necessary to identify and handle a data leakage incident (such as 'what information is sensitive', 'how is reference information stored'). In this section we look at how such an implementation could look like.

Comparing information

Regardless of the information detection method there has to be a way to compare the information being transferred with information that should not be transferred. If there is a central storage location for all sensitive files the DLP solution can compare captured information with the stored files. When the information is more distributed there has to be some method to retrieve the information (or a unique identifier of that information) and import it to the DLP system or inform the DPL system of the location of the information. An example of this is agents running on user computers scanning files for sensitive information and reporting the location and contents of sensitive information to a central DLP control point.

Instead of copying full files to the database of the DLP system there has to be some consideration of what to compare. This is the classification process; what makes certain information sensitive (refer to the previous section). This, and the accompanied 'How would you distinguish this information in data traffic', is the required information the DLP needs for monitoring and detecting information leaks.

Much information is transferred via files. Files have some identifiers that when used together can be used to uniquely identify it. The DLP system can take advantage of this property. There are many file characteristics and meta-data information that can be used. Some of these depend on the file type while some are available for all files. File characteristics that can be used include;

- MD5 hash of the file;
- Filename;
- Object name;
- Filesize;
- Editor properties:
  - Author;
  - Last modified by;
  - Company / Manager;
- Number of pages / words;
- Total editing time;
- Tags / Keywords;
- Comments;
- Template used;
- Status;
- Date information:
  - Creation date;
  - Last modification date;
  - Last printed.
- Copyright Notice;
- Credit;
- Original Transmission Reference;
- Special Instructions.

This information, in conjuction with the information itself, is very helpful in determining the whether a transferred file is equal to another (sensitive) file.

In not all cases (original) files can be compared; captured traffic can include (portions of) sensitive information that is not kept in its original format. For such information types (exact) file comparison is not possible and many of the file characteristics do not apply. In this case, the comparison can only be performed on (portions of) the captured information itself and the known context of the sender and receiver (this is, of course, also possible when comparing files). The same holds for information that is not transferred via files.

Contextual information that can prove to be useful is information regarding the sender / receiver:

- User- / Accountname;
- IP address;
- Geographical location;
- Time of sending / Time of receiving;
- Software used;
- Method of sending.

Some classification / detection techniques can be used both to match files and to match information transferred not via files (such as rule-based detection, regular expressions, database fingerprinting, partial document matching) while exact file matching will only work on the first.

Similarity

In many cases not all of the aforementioned file- and transfer/channel characteristics will be available at all times, but the combination of all applicable content and contextual information regarding an information transfer can be used to determine how similar a transfer is to a data leakage incident. This similarity is key in detecting incidents and separating regular traffic from a data leak.

This similarity can be computed in different ways (such as Mahalanobis distance, Pearson correlation, cosine similarity, or kernel functions) but no matter how the similarity of one data instance (one characteristic) with another one is computed, the difficulty lies in relating certain combinations of similarities to incidents and others to regular traffic. An example of a possible incident could be that the combination of a certain time, sender location, and use of transfer method could indicate a possible incident, while that same combination on another time does not. The more characteristics are used, the more detailed and difficult the comparison becomes.

In a way, the machine learning ICMP Proof-of-Concept as shown in section B follows the same approach; different data streams are compared to each other and a new (and unknown) input is classified as the data type it most resembles (either real packet or fake packet).

The similarity of the information itself is the most important characteristic of a data transfer; if an outgoing file transfer contains information similar to a sensitive document other characteristics (such as time or transfer method) do not matter much any more.

When transferring information via covert channels this most important characteristic (the information itself) is hidden. In this case, other characteristics (such as sender/receiver IP and geographical location) play a larger role.

Linking information

All this information regarding sensitive information, characteristics, and rules needs to be stored somehow. This information then needs to be linked together.

One way of doing this is storing the information regarding the sensitive files in a database and querying it using a database query language such as SQL. Such a database should contain information regarding the sensitive information itself, the captured datastreams, incidents, and responses. The contents of sensitive information can include the information itself but also regular expressions or hashes of files.

An example database layout is given in Figure 7.1. In this figure, a datastream (captured transfer) has a content and context field. If applicable, the contents can have characteristics. Incidents are stored with their time of occurrence, associated datastream and information, and response. The sensitive files themselves are stored, like datastreams, as the information itself and a series of characteristics (which might not all apply to all information). Added is the notion of sensitivity.

With this information a database query language can be used to determine if a DLP incident occured.

An example in pseudocode:

```
# A new datastream was captured
datastream = SELECT Content, Context FROM Datastream WHERE ID = 'latest';

match = SELECT * FROM SensitiveInformation WHERE SensitiveInformation.Content LIKE datastream.Content;
match2 = SELECT * FROM SensitiveInformation WHERE SensitiveInformation.Context LIKE datastream.Context;

if match or match2:
  respond();
else:
  no_incident();
```

The above snippet only responds when content or context between sensitive information and captured datastream

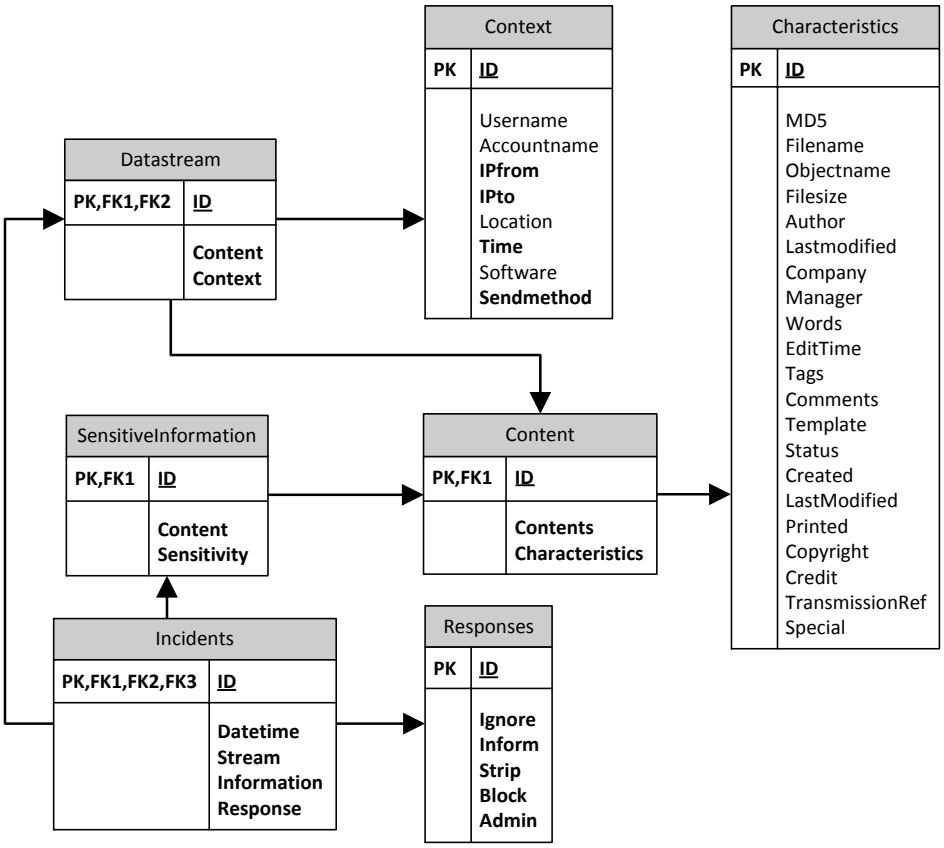


Figure 7.1: Concept database layout.

are exactly alike but this is no requirement; it is possible to respond when only certain fields match. An example:

```
match2 = SELECT * FROM SensitiveInformation WHERE
         SensitiveInformation.Content.Characteristics.MD5 = datastream.Content.Characteristics.MD5
         AND
         SensitiveInformation.Context.IPfrom = datastream.Context.IPfrom;
```

Another option is creating 'thresholds' such as; respond only when at least ten characteristics are the same and the content is at least 50% similar.

Response

After detection of an incident it needs to be handled. There are several matters that need to be taken into account, the two most important ones being:

- When is a DLP violation responded to; immediately, after several times the same incident, or after several times the same user.
- How to respond; Inform, strip file from transfer (if applicable), block, and/or notify the administrator.

A DLP violation should be responded to immediately when it is obvious that a severe incident has happened or is about to take place. Some cases however might not be so clear; an individual DLP violation might be an outlier, false

positive, or minor incident. A series of these violations however might indicate a more serious DLP incident. A trigger can be the number of similar incidents or the number of different incidents to (or from) the same user. Capturing also the minor and often overlooked incidents can also help in detecting covert channels which often only transfer small amounts of information and thus largely trigger only small incidents or oddities in monitoring software.

The policy should contain information on when to consider a DLP violation severe enough to trigger a response. This depends (among other things) on the sensitivity of the information, the method of transfer, and the sending or receiving party.

After the decision has been made to respond there are multiple ways to react depending on the severity of the data leakage incident. In case of a clear data leakage incident in which the information being transferred is highly sensitive (e.g., a blueprint of a new product) the transfer should be blocked, the administrator notified, and the incident

Concept

An example in pseudo-code that gets the information from a detected file (if applicable) and which processes the capture is given below:

```
#Main function. Start here.
main() {
  #datastream is the raw datastream
  global import datastream

  #context contains additional information such as sender/receiver ip, time, etc.
  global import context

  similarity = new list;

  classify(similarity)
  if similarity[critical] > threshold:
    respond(severe);
  if similarity[medium] > threshold:
    respond(medium);
  if similarity[low] > threshold:
    respond(low);
}


classify(similarity) {
  # Patterns that contain content that is very likely to indicate a data leakage event
  # Example pattern Credit cards: \b4[0-9]{12}(?:[0-9]{3})?\b
  import patterns_critical;

  # Patterns that contain content that is likely to indicate a data leakage event.
  # Example pattern phone number: \b3[0-9]{6}\b
  import patterns_medium;

  # Patterns that contain content that on its own does not indicate a data leakage
  # event but when combined might. Example pattern employee number: \b6[0-9]\b
  import patterns_low;

  similarity[critical] = similarity(patterns_critical);
  similarity[medium] = similarity(patterns_medium);
  similarity[low] = similarity(patterns_low);
}

similarity(pattern) {
  if match(pattern, critical) {
```

```
    respond(critical);
  }

  if match(pattern, medium) {
    send_warning_to_user();
    add_incident(context);
    respond(medium);
  }

  if match(pattern, low) {
    add_incident(context);
    respond(medium);
  }
}

respond(severity) {
  # Respond immediately: Block and inform administrator
  if match(severity, critical) {
    block();
    notify();
  }

  # Block transfer only.
  if match(severity, medium) {
    block();
  }

  # Check whether this sender has been in several (low) incidents.
  #If true, block, else consider it a false positive.
  if match(severity, low) {
    #if the sender has been in several low events, suspect a data leakage situation
    if (sender_incidents > limit)
      respond;
    else
      ignore;
  }
}
```

Chapter 8

Theory & Practice

Now that we have created a model for DLP via covert channels we will compare our theory with an existing product available in the market today; Symantec Data Loss Prevention 11. To do this we will look at what products Symantec's solution consists of and how the solution works. We will then evaluate to what extend the product employs our DLP model.

8.1 Symantec DLP 11 Overview

Symantec has one of the best Content-Aware Data Loss Prevention products in the market today; According to Gartner's 'Magic Quadrant for Content-Aware Data Loss Prevention' Symantec leads with their completeness of vision and the ability to execute. Vendors in the leaders quadrant 'have demonstrated good understanding of client needs and offer comprehensive capabilities in all three functional areas - network, discovery, and endpoint - directly or through well-established partnerships and tight integration' (21) (See Figure 8.1).

Figure 8.1: Symantec leading the Magic Quadrant (Source: Gartner (21))

Symantec's DLP 11 solution consists of eleven products divided over three domains; Network, Storage, and Endpoint.

In the Network domain there are products to monitor and prevent data leakage over the network. The three products in this domain are Network Monitor, Network Prevent E-mail, and Network Prevent.Web. The first of these products can only be used to inspect traffic, while the latter two can also block unwanted traffic.

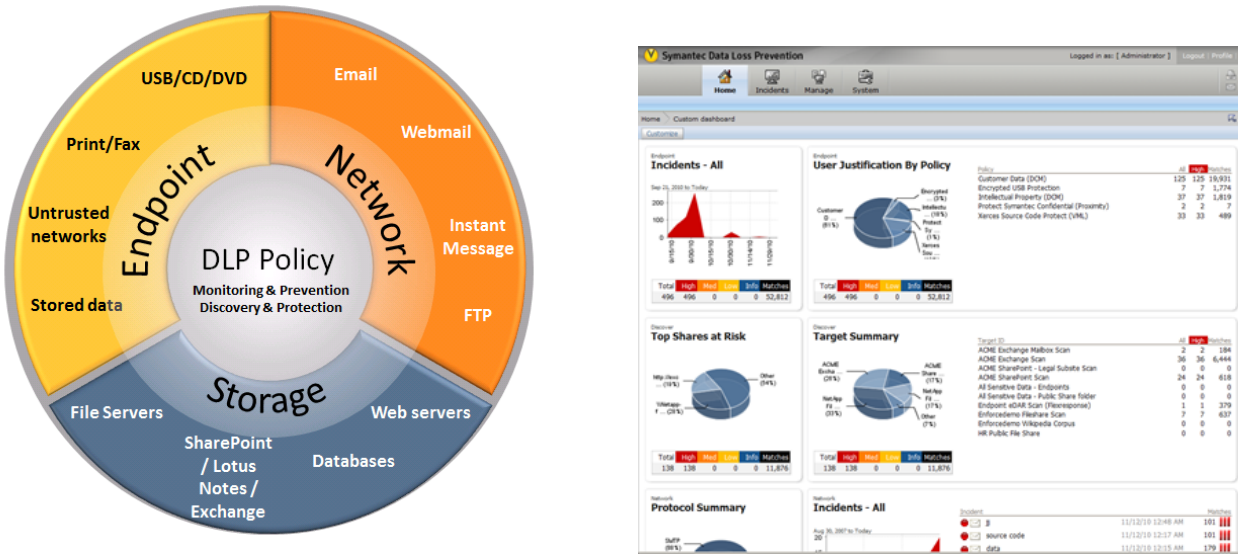


Figure 8.2: Symantec DLP domains and Home screen of Symantec's DLP 11 Enforce server.

The Storage domain also contains three products; Network Discover, Data Insight Enterprise, and Network Protect. Network Discover is used to find where sensitive data is stored within the network (e.g., shared drives, databases). The Data Insight Enterprise product provides information on who used what information at a certain time. Network Protect is used to enforce the policy when data leakage appears to be appearing.

There are only two products in the Endpoint domain; Endpoint Discovery and Endpoint Prevent. Endpoint Discovery locates possibly sensitive information on the individual endpoints, Endpoint Protect is used to prevent sensitive information from leaving the Endpoint (e.g., via USB-sticks or via printer).

The central control mechanism to these components is provided by the last Symantec DLP product; the Enforce Server. All policies are created and maintained via the Symantec Enforce Server. All other Symantec DLP products communicate with this server for policy updates, reporting, and for configuration changes.

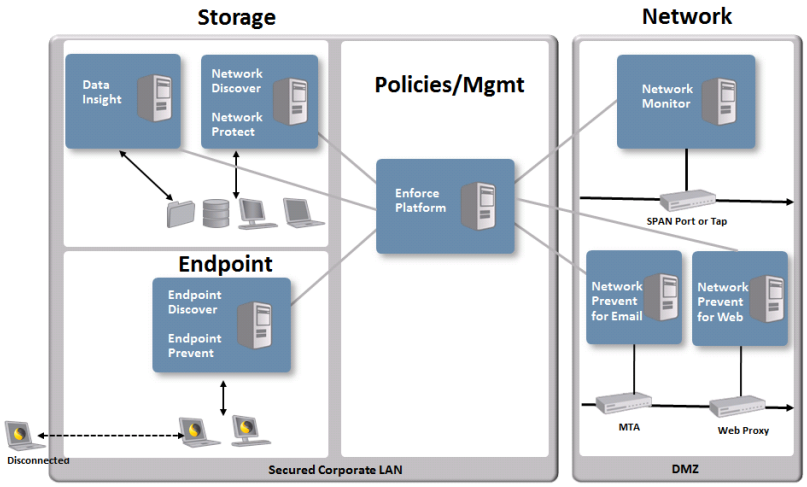


Figure 8.3: Symantec DLP Architecture (Source: Symantec (59))

8.2 Model compared to Symantec DLP 11

In our model we assume the existence of a way to enforce a security policy (referred to as the 'DLP system'). Sending of information goes through this central system. Our model employs several channel inspection- and modification techniques (such as encryption, channel analysis, deep content analysis), policy rules to abide by (e.g., no storing of sensitive information on publicly accessible shared resources), and possible responses to occurrences (e.g., blocking, notifications, logging).

In this section we compare the model with Symantec's DLP product to see to what extend, and how, it implements the model, and whether or not the discussed deep content analysis methods are implemented.

Symantec and the different scenarios In chapter 6.1 several scenarios are mentioned. To recall:

1. Sending sensitive information from an insider to an outsider directly;
2. An outsider accessing sensitive information from a shared resource;
3. an insider accessing sensitive information from a shared resource;
4. an insider sending sensitive information directly to another insider;
5. Sending non-sensitive information from an insider to an outsider directly.

	Policies
Access to policies	Through the Enforce Server policies can be configured
Policy storage	The policies which are created and configured on the Enforce server are stored at the Enforce server itself but copies are placed at the different components to allow them to function even when the Enforce server is off-line.
Enforces policies (allow, block, notify, ...)	Policies are enforced at the individual components. Each policy is build-up from three components; the detection rule, the group rule, and the response rule ('what' to detect, 'who' to monitor, and 'how' to react respectively)
	Information flow
All information flows through a central 'DLP System'	No, information flows through/by individual components; only reports about an incident are reported to Enforce.
All communications carrying sensitive information should be encrypted	Policies can be configured to block communications carrying sensitive information on specific insecure protocols. Possibility to intercept sensitive documents and encrypt it before transferring it to the final destination is not supported.
Secure information sending has to go through secure channels	As the above. Possibility to intercept sensitive documents and automatically forward the information on a secure channel is not possible.
Channels using a certain protocol can be inspected for 'strange behaviour'	Symantec DLP has the possibility to inspect specific protocols for oddities; this can be configured through policies.
	DCA techniques
Rule-based	Yes, this is one of the prime methods to detect information. Word matching, and combinations of words, IP address matching, username matching, and e-mail address matching are some of the things that can be used to create policy rules to which the system should react.
Regular expressions	Yes, for example credit card numbers or driving license numbers.
Database fingerprinting	No
Exact file matching	Yes, for example the sending of known classified files via e-mail can be detected.
Partial document matching	Yes, a policy can match on a part of information.
Statistical analysis	No

Conceptual/lexicon analysis	Yes, policies can for example be configured to only trigger an information leakage incident if a certain group of words is detected, or when a threshold of occurrences of a certain word is reached. The word 'important' might not be flagged while 'important' in combination with other words 'blueprint', 'latest model', and 'highly confidential' does raise an incident.
Category matching	Symantec DLP 11 contains a large set of pre-built policies specifically targeted at regulatory compliance. Examples include PCI, HIPAA, and SOX.
Searching in encrypted data	No
Profiling	There is a possibility to monitor incidents which occur from a certain IP address, username, or e-mail address.
Machine Learning	Yes, machine learning is done through Support Vector Machines.
	Perimeters
File perimeter	Yes, creation of files by copying sensitive information from one file to another can be prevented using Endpoint protection.
Machine perimeter	Yes, using Endpoint protection.
Department perimeter	Yes, using Network Monitor and Network Prevent
Company perimeter	Yes, using Network Monitor and Network Prevent
	Reporting
Users can be notified	Yes

Table 8.1: Comparison Symantec DLP and DLP model

Symantec's DLP product provides a varied set of information detection algorithms; besides basic rule-based detection and regular expressions a more advanced detection technique (Vector Machine Learning) is also employed. Searching can be limited to only certain (groups of) people (based on IP address, username, e-mail address, etc.), communication channels (filter on protocol), and custom rules on how to react can be created, including showing messages to 'information leakers' (in combination with Endpoint Protection) and blocking information from reaching its destination.

Some techniques that are discussed in this paper but are not supported by Symantec are encryption of all channels and the more advanced detection / classification algorithms.

Different focus area
The factor 'covert channels' is not an important one for the Symantec product; their focus is aimed at benevolent employees who mistakenly or out of lack of other options ('I send this file to my personal e-mail to work at home because there is no VPN option available') cause data leakage events while the DLP model and the CC model in this paper are designed with malicious intent in mind.

Chapter 9

Related Work

It is important to know and understand what others have done in Data Leakage Prevention related fields and in the area of Covert Channel detection and -prevention. In this chapter an overview is given of some of the related work.

***

9.1 Related work: Data Leakage Prevention.

In an attempt to prevent data leakage from a decision theoretic approach Marecki et al. formulate the data leakage prevention problem as Partially Observable Markov Decision Processes (POMDP) in their paper `**A Decision Theoretic Approach to Data Leakage Prevention**' (37). In their model they encode a monitoring mechanism -digital watermarking- invisible for the recipients. Their goal was to derive optimal information sharing strategies for the sender and optimal information leakage strategies for a malicious recipient (leaking information results in a positive reward for the recipient and a penalty to the sender). An assumption is that the sender can detect information leakage (imperfect, through the digital watermarking). Their study starts with one recipient who either leaks out all packets is receives from the sender or none of them. Next, they consider a 'fuzzy' recipient leaks only a certain percentage of the packets it receives. Finally they generalize their model where the sender shares information with multiple fuzzy recipients. Through experimentation they show the effects of different settings for leak cost, initial trust a sender has in his recipients, number of epochs and the number of fizziness levels. The efficiency of their system degrades gracefully with the efficiency of the underlying monitoring mechanism. They believe their approach is the first to address complex information sharing under uncertainty.

In `**A Model for Data Leakage Detection**' (49) and `**Data Leakage Detection**' (50) the authors address the problem of sharing sensitive data between a distributor and supposedly trusted agents (third parties). The authors claim that when some of the data is leaked their proposed data allocation strategy improves the probability of identifying leakages without relying on alterations of the data, such as watermarks. Their identification method is based on the overlap of each agent's data with the leaked data, the data of other agents, and the possibility to guess the data. The basic idea is that by computing the probability that an agent leaked the data for all agents, the agent with the highest probability is most likely to have leaked the data. By injecting fake data records the detection rate improves even further, according to Papadimitriou et al. The research is useful for when a data breach has been occurred; depending on the situation within an enterprise the (relatively simple) method described in the paper can be used to detect the source of the leakage. A requirement that is made in the paper (`it is known who sent what to whom') might not apply in the situation of large corporations where lots of information is transferred between employees, customers, suppliers, and other parties.

The researchers Yokomori et al. create a security analysis implementation method to certify the security of a program by statically analyzing the information flow in their paper `**Analysis and implementation method of program to detect inappropriate information leak**' (66). Their work is based on theoretic work by Kuninobu et al. (35). Kuninobu et al. use a program analyses technique based on a lattice model of security classes to guarantee that a given

program never leaks secret information to insecure storage. Yokomori et al. use the theory from Kuninobu et al. to construct an implementation and extent his theory to include global variables and procedural calls. They validate their implementation by applying the constructed tool to a simple credit card program. This technique and the created implementation focus on the time of software construction unlike our thesis which focuses on the situation where applications are already deployed within an enterprise.

In the paper '**State of the Art in Network-Related Extrusion Prevention Systems**' (26) written in January 2009, the authors give an overview about some 'state-of-the-art' network related 'extrusion prevention systems' (a synonym for data leakage prevention systems). They describe some of the techniques used in Network-Related Systems. Network-Related systems consist of several system parts such as a network monitor, e-mail integration capabilities, filtering/blocking and proxy integration. They list seven content analysis techniques as mentioned in the **Global Data Leakage Report 2009** by Infowatch (43); rule-based/regular expressions, database fingerprinting, exact file matching, partial document matching, statistical analysis, conceptual/lexicon, and categories. Some existing security solutions include network layer firewalls, application layer firewalls, proxies, and Intrusion Detection/Prevention Systems (IDS/IPS). They mention some weaknesses in their paper; the lack of encrypted data or encrypted connections, transport of application protocols across unusual channels, and possible violation of data privacy laws. This paper is helpful in our research because of the different analysis techniques they mention which are helpful in showing how Deep Content Analysis works in section 5.6.

'**Detecting Insider Theft of Trade Secrets**' (11) by a group of MITRE researchers provides new insight into the behaviour of malicious insiders. They identify four essential aspects of detecting insider threats: The first is that one must monitor the activities necessary for trusted insiders to use information. Second is the observation that this monitoring process must occur at the application layer. Third is the observation that contextual information is also needed (both 'user context' and 'information context'). Fourth is the requirement for methods to fuse alerts and rank individuals for analysts to review. The researchers performed an empirical study of malicious behaviour to improve their understanding of user behaviour by dividing a test group of fifty people in a malicious and a benign group. Both had to complete an information search task. They conclude by saying that although their preliminary analyses revealed interesting and significant patterns in malicious behaviour, they have not identified any one type of behaviour that distinguishes malicious users from benign ones. They mention that the most valuable way to tackle insider threats seems to be casting a wide net over many behaviour patterns and evaluate those. In our paper we create a model for DLP but do not provide a blueprint for setting up a DLP implementation. This paper is useful for DLP system vendors to make their solutions perform better.

Spitzner applies the technology of honeypots to detect, identify, and gather information on malicious insiders in his paper '**Honeypots: Catching the Insider Threat**' (58). A honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource; anything or anyone interacting with a honeypot is an anomaly. A difference with a regular honeypot setup is that a honeypot with the goal to catch an insider must be placed within the internal network, and it has to 'advertise' itself. The author states that to learn more about the attacker a more sophisticated honeypot has to be used. The author concludes his paper by noting that honeypots can contribute to the early detection of advanced insider threats. Our model does not use honeypots.

'**Preventing Information Leaks through Shadow Executions**' by Capizzi et al. addresses the concern of information confidentiality of end-users. The approach they use is called 'shadow execution'. Shadow execution consists of replacing the original application with two copies of the same program that run the same code but are initialized with different sets of inputs, and different restrictions are imposed. One copy can access the Internet using 'fake' input (the 'public copy'), the other copy cannot access the Internet but uses the user's data as input (the 'private copy'). The response obtained from the public copy can then be used by the private copy. A key property of shadow execution is that it allows applications to successfully communicate over the network while disallowing any information leaks.

9.2 Related Work: Covert Channels.

In '**Covert channel capacity**' (41) Millen establishes a connection between Shannon's theory of communication and information flow models, such as the Goguen-Meseguer model, that view a reference monitor as a state-transition automaton. Information flow models attempt to explain all possible ways in which information may be compromised in a computer system. Some models share the philosophy that information flow in a computer security context is related to inference: if one user can, by observing outputs available to him, deduce something about inputs from another user, there has been some information flow. Other models are automata-based information

flow models, such as the Goguen-Meseguer model, which regard secure systems as automata or state-transition machines. A problem with such models is that they do not measure the rate of information flow, but only whether it exists or not. In the paper, they define a perfect channel as a system with one input ('X') and one output ('Y'), in which the output is only determined functionally by the value of the input. Because a real channel is usually noisy, Y is not determined solely by X, but instead has a probability distribution determined by the characteristics of the channel. The information flow in 'bits per trial' between users X and Y is defined as the capacity of the channel, calculated under the assumption that inputs from X are independent of those from other users. A trial is a period of time between outputs to Y. It has been shown that if X is non-interfering with Y, then the information flow is zero. However, the information flow can be zero even when X is not non-interfering with Y. In the paper they show that non-interference implies that the information flow is zero, but that it can indicate the (possible) existence of information flow in cases where no information flow actually exists because it does not take into account the possibility of independent users.

Bidou et al. focus on the concept of covert channels and provide examples of application and detailing advantages and drawbacks in their paper '**Covert Channels**' (7). Starting with the introduction of covert channels in 1973 by Lamson (36), he describes covert channels characteristics; capacity, noise, and transmission mode. A section on basic techniques mentions some Network Channel drawbacks; using the IPID field in the IP Header ('capacity of 2 bytes is useless in terms of data transfer'), fields in the UDP Header ('no guarantee on reliability and integrity of the communication'), and fields in the TCP Header ('anomaly detection engines have a huge impact on the available load', protection mechanisms such as SYNCookies renders the use of some fields impossible). A drawback for Application Channels is the fact that behaviour analysis will make it possible to detect such channels. The paper ends with describing several 'bouncing' techniques such as SYN/ACK bounce, ICMP Error bounce, Application Layer bounces, and Multiplexing bounces. In chapter 4.2 we also take a look at Covert Channel attacks.

The paper '**Covert Data Storage Channel Using IP Packet Headers**' by Thyer investigates the IPID header fields of the Internet Protocol (IP) (61). RFC-791 documents the proper use of the Internet Protocol Identification (IPID) field; it does, however, not document the IPID field values within the context of non-fragmented traffic, or any reference to IPID initialized values, or usage within the context of different layer 4 protocols. During kernel development of Operating Systems the designers and implementers choose the 'exception case' uses of the IPID field themselves. Different Operating Systems had differences in implementation and differences of fragmentation reassembly and non-fragmented packet reception/transmission have allowed for mechanisms for Operating System fingerprinting, stealth network scanning, denial of service, and covert data transmission. The paper gives an overview of covert channels, examines various fields of the IP header that can potentially be exploited for covert transmission, documents a practical storage covert channel software implementation with detailed packet analysis, summarizes implementation challenges, and concludes with potential detection and prevention techniques. In chapter 4.2 we also take a look at Covert Channel attacks.

Berk et al. investigate the channel capacity of Internet-based timing channels and propose a methodology for detecting covert timing channels based on how close a source comes to achieving that channel capacity in their paper '**Detection of Covert Channel Encoding in Network Packet Delays**' (5). The data leaking mechanism focused upon in this paper is the use of delay times between packets to encode data. This means that the intruder does not necessarily have to create new traffic; the time between packets of regular communications is merely modulated to encode data. This covert channel is used as an example in chapter 4.2.

'**When Insiders Attack: Covert Data Channels**' by Taylor J.B. deals with the insider threat; an insider with legitimate access to information and technically skilled enough to create covert data channel (60). The paper explores statistics about actual insider thefts, the problem of confinement and multiple types of non-standard data channels. The covert timing channel is focused upon later in the paper. In our paper we focus on the insider threat as well.

Steganography in covert channels is the main point of interest in a paper by Owens titled '**A Discussion of Covert Channels and Steganography**' (48). Owens uses the definition from the paper from Katzenbeisser et al. '**Information Hiding Techniques for Steganography and Digital Watermarking**'; '*While cryptography is about protecting the content of messages, steganography is about concealing their very existence. (Steganography) is usually interpreted to mean hiding information in other information*' (32). He categorizes steganography methods in three distinct modes: injection (adding custom data to existing data), substitution (substituting data within data) and propagation (generating a new file). After giving steganography examples of TCP/IP Stego and Audio & Video Steganograms the paper discusses Steganalysis; the process of investigation to determine the presence of a steganographic payload. Statistical analysis of the digital content of suspected steganograms provides the best means of detection, he mentions. The objective would be to determine if the file's statistical properties depart substantially enough from a norm to make it suspicious. Determining the norm is a difficulty; one has to look at footprints and expected patterns. The author's stance on detection and reaction mechanism for protecting against steganography is '*Monitor those things*

you wish to protect'. In our paper we consider steganography to be a subliminal covert channel.

Bottcher et al. wrote an article for the International Conference on Availability, Reliability and Security in 2008 titled `**Detecting suspicious relational database queries**' (9). They present a forensic approach to privacy violation control that after information has been leaked identifies who had access to the leaked information by representing secret information as a Boolean formula and comparing it with the queries and the database in order to identify suspicious queries. By `suspicious queries' they refer to queries that have got sufficient information to infer secret information that has been leaked.

Chapter 10

Conclusion

The original research question was 'How can covert channel data leakage be detected and prevented?'.

The answer to the detection part of this question is that it is fairly difficult to detect data leakage via covert channels mainly because of the difficulty to detect the covert channels themselves. In our model we apply covert channel detection mechanisms including Channel Analysis and Deep Content Analysis. These techniques can be used to detect covert channels within communication channels and hidden information within other (non-sensitive) information.

Preventing covert channel data leakage is started by preventing the existence of covert channels; only allowing information to be sent through known and secure channels which are approved by the security policy. By combining what we know about covert channels and Information Flow models our model tries to prevent the existence of covert channels in the first place. By also applying the DLP techniques between users within the same enterprise we try to prevent sensitive information from flowing 'down' to parties with less clearance (the Bell-LaPadula model) and between users who have a conflict of interest (Chinese Wall model).

The heart of the DLP model, the Security Policy, is essential in any DLP solution implementation. It contains information on who has access to what, who can send information to where, where conflicts of interest might arise, and more.

The actual detection of sensitive information and covert channels is done by Deep Content Analysis techniques and Channel Analysis methods. Traditional Deep Content Analysis such as rule-based/regular expressions, database fingerprinting, exact file matching, partial document matching, statistical analysis, conceptual/lexicon analyses, and category matching are useful and already available in several products. Experiments and market leading products are already trying out new techniques such as searching in encrypted date, profiling, and machine learning which further enhance the potential of DCA and improve the overall success of DLP solutions. Channel Analysis, inspecting transfer channels for oddities, increases the chance of finding hidden data streams within existing streams.

To conclude, covert channel data leakage prevention is possible but not likely to be bulletproof (Refer to 'Discussion' below). Most techniques are all there; SSL for secure traffic is widely used, encryption techniques for secure storage is already 'best practice' for almost any type of sensitive information, Deep Content Analysis and Channel Analysis methods are no longer merely of academic interest, and traditional protection mechanisms (Firewalls/Proxies/IDS/IPS) are deployed in any serious organization.

In this research we've created a model for both DLP and covert channel DLP. With these models we hope to achieve that the various discussed techniques can be applied in such a way as to gain a better protection of the sensitive documents within organizations.

Adam Cornelissen

Chapter 11

Discussion

Our study shows that it is possible to create a model for Data Leakage Prevention and Covert Channel information leakage.

The fact that it is possible to create a model does not automatically imply that it is always possible to stop data leakage; 100 percent data leakage prevention is practically impossible. Simply having access to sensitive information opens the possibility to leakage; one can remember the contents of the information and write it down at home. Most current DLP solutions focus on the 'low hanging fruit', the solutions with the best 'low-cost/high-result' ratio that can prevent data leakage, e.g. notification when trying to send sensitive information. DLP systems that focus on employees intentionally smuggling out information to people outside of the organization specifically are non-existent. The never ending resourcefulness of the human being will likely allow a determined, knowledgeable and sufficiently financed employee to leak information, no matter how much a company disallows, blocks, or tries to detect leakage of information. The mantra here would be that the scope and level of protection should be specific and appropriate to the asset at risk (Jericho 1st commandment, see A)

Another problem with data leakage prevention systems is the false-positives problem. Especially with systems that employ Deep Content Analysis or Channel Analysis this would be a problem because the is always the chance of drawing the wrong conclusion from an observation (e.g., wrongly classifying/detecting information, or misjudging an anomaly for a leakage event).

The more information flowing over a covert channel resembles legal traffic allowed by the security policy, the more difficult it is to detect until the point that it becomes almost impossible to determine.

Promising behaviour- or traffic analysis methods that adapt and learn from users can be influenced by slowly 'training' irregular behaviour to become 'normal' behaviour. Again, the question regarding when it is no longer viable to perform an attack has to be posed.

Appendix

Appendix A

Jericho Commandments

Jericho Forum™ Commandments

The Jericho Forum commandments define both the areas and the principles that must be observed when planning for a de-perimeterized future.

Whilst building on "good security", the commandments specifically address those areas of security that are necessary to deliver a de-perimeterized vision.

The commandments serve as a benchmark by which concepts, solutions, standards, and systems can be assessed and measured.

Fundamentals

1. The scope and level of protection should be specific and appropriate to the asset at risk.

- Business demands that security enables business agility and is cost-effective.
- Whereas boundary firewalls may continue to provide basic network protection, individual systems and data will need to be capable of protecting themselves.
- In general, it's easier to protect an asset the closer protection is provided.

2. Security mechanisms must be pervasive, simple, scalable, and easy to manage.

- Unnecessary complexity is a threat to good security.
- Coherent security principles are required which span all tiers of the architecture.
- Security mechanisms must scale; from small objects to large objects.
- To be both simple and scalable, interoperable security "building blocks" need to be capable of being combined to provide the required security mechanisms.

3. Assume context at your peril.

- Security solutions designed for one environment may not be transferable to work in another. Thus, it is important to understand the limitations of any security solution.
- Problems, limitations, and issues can come from a variety of sources, including geographic, legal, technical, acceptability of risk, etc.

Surviving in a Hostile World

4. Devices and applications must communicate using open, secure protocols.

- Security through obscurity is a flawed assumption – secure protocols demand open peer review to provide robust assessment and thus wide acceptance and use.
- The security requirements of confidentiality, integrity, and availability (reliability) should be assessed and built in to protocols as appropriate; not added on.
- Encrypted encapsulation should only be used when appropriate and does not solve everything.

5. All devices must be capable of maintaining their security policy on an un-trusted network.

- A "security policy" defines the rules with regard to the protection of the asset.
- Rules must be complete with respect to an arbitrary context.
- Any implementation must be capable of surviving on the raw Internet; e.g., will not break on any input.

The Need for Trust

6. All people, processes, and technology must have declared and transparent levels of trust for any transaction to take place.

- Trust in this context is establishing understanding between contracting parties to conduct a transaction, and the obligations this assigns on each party involved.
- Trust models must encompass people/organizations and devices/infrastructure.
- Trust level may vary by location, transaction type, user role, and transactional risk.

7. Mutual trust assurance levels must be determinable.

- Devices and users must be capable of appropriate levels of (mutual) authentication for accessing systems and data.
- Authentication and authorization frameworks must support the trust model.

Identity, Management, and Federation

8. Authentication, authorization, and accountability must interoperate/exchange outside of your locus/area of control.

- People/systems must be able to manage permissions of resources and rights of users they don't control.
- There must be capability of trusting an organization, which can authenticate individuals or groups, thus eliminating the need to create separate identities.
- In principle, only one instance of person/system/identity may exist, but privacy necessitates the support for multiple instances, or one instance with multiple facets.
- Systems must be able to pass on security credentials/assertions.
- Multiple loci (areas) of control must be supported.

Access to Data

9. Access to data should be controlled by security attributes of the data itself.

- Attributes can be held within the data (DRM/metadata) or could be a separate system.
- Access/security could be implemented by encryption.
- Some data may have "public, non-confidential" attributes.
- Access and access rights have a temporal component.

10. Data privacy (and security of any asset of sufficiently high value) requires a segregation of duties/privileges.

- Permissions, keys, privileges, etc. must ultimately fall under independent control, or there will always be a weakest link at the top of the chain of trust.
- Administrator access must also be subject to these controls.

11. By default, data must be appropriately secured when stored, in transit, and in use.

- Removing the default must be a conscious act.
- High security should not be enforced for everything; "appropriate" implies varying levels with potentially some data not secured at all.

Conclusion

De-perimeterization has happened, is happening, and is inevitable; central protection is decreasing in effectiveness:

- It will happen in your corporate lifetime.
- Therefore, you need to plan for it and should have a roadmap of how to get there.
- The Jericho Forum has a generic roadmap to assist in the planning.

Appendix B

PoC: ICMP covert channel

The experiment was performed between a Windows XP machine and a Windows 7 machine (send and receive respectively). In total 1363 packets were used (of which 241 'malicious'). The 'malicious packets' were made with ICMPForge (28). The test used the Fast Artificial Neural Network Library (FANN), a free and open-source neural network library (45). Compile using 'gcc train.c -o train -ldoublefann -lm' and 'gcc test.c -o test -ldoublefann -lm'.

Neural Network training source code:

```
#include "doublefann.h"
int main()
{
const unsigned int num_input = 19;
const unsigned int num_output = 1;
const unsigned int num_layers = 3;
const unsigned int num_neurons_hidden = 25;
const float desired_error = (const float) 0.03;
const unsigned int max_epochs = 500000;
const unsigned int epochs_between_reports = 1;
struct fann *ann = fann_create_standard(num_layers, num_input,
num_neurons_hidden, num_output);
fann_set_activation_function_hidden(ann, FANN_SIGMOID_SYMMETRIC);
fann_set_activation_function_output(ann, FANN_SIGMOID_SYMMETRIC);
fann_train_on_file(ann, "input.data", max_epochs, epochs_between_reports, desired_error);
fann_save(ann, "neural.net");
fann_destroy(ann);
return 0;
}
```

Neural Network testing source code:

```
#include <stdio.h>
#include "doublefann.h"

int main()
{
fann_type *calc_out;
unsigned int i;
int ret = 0;

struct fann *ann;
struct fann_train_data *data;
```

```
printf("Creating network.\n");
ann = fann_create_from_file("neural.net");

if(!ann) {
printf("Error creating ann --- ABORTING.\n");
return 0;
}

printf("Testing network.\n");
data = fann_read_train_from_file("try.data");

for(i = 0; i < fann_length_train_data(data); i++) {
fann_reset_MSE(ann);

calc_out = fann_test(ann, data->input[i], data->output[i]);

printf("Test (%f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f, %f)
        -> %f, should be %f, difference=%f\n\n",
        data->input[i][0],  data->input[i][1],  data->input[i][2],  data->input[i][3],
        data->input[i][4],  data->input[i][5],  data->input[i][6],  data->input[i][7],
        data->input[i][8],  data->input[i][9],  data->input[i][10], data->input[i][11],
        data->input[i][12], data->input[i][13], data->input[i][14], data->input[i][15],
        data->input[i][16], data->input[i][17], data->input[i][18],
        calc_out[0], data->output[i][0], (float) fann_abs(calc_out[0] - data->output[i][0]));
}

fann_destroy_train(data);
fann_destroy(ann);

return ret;
}
```

Statement of Originality

I herewith declare that I have produced this paper without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This paper has not previously been presented in identical or similar form to any other Dutch or foreign examination board.

The thesis work was conducted from October 1, 2010 to June 25, 2012 under the supervision of Martijn Knuiman and Derk Wieringa at Deloitte Risk Services, and Bart Jacobs at the Radboud University Nijmegen.

Nijmegen/Amstelveen, June 25, 2012

Bibliography

(1) A. Aamodt and M. Nygård. Different roles and mutual dependencies of data, information, and knowledge–An AI perspective on their integration. *Data & Knowledge Engineering*, 16(3):191–222, 1995.

(2) R. Accorsi and C. Wonnemann. Auditing workflow executions against dataflow policies. In *Business Information Systems*, pages 207–217. Springer, 2010.

(3) R.L. Ackoff. From data to wisdom. *Journal of Applied Systems Analysis*, 16(1):3–9, 1989.

(4) D.E. Bell, L.J. La Padula, and MITRE CORP BEDFORD MA. Secure computer system: Unified exposition and Multics interpretation. 1976.

(5) V. Berk, A. Giani, G. Cybenko, and NH Hanover. Detection of covert channel encoding in network packet delays. *Department of Computer Science, Dartmouth College, Technical Report TR2005536*, 2005.

(6) E. Bertino. *Computer security, ESORICS 96: 4th European Symposium on Research in Computer Security, Rome, Italy, September 25-27, 1996 : proceedings*. Lecture notes in computer science. Springer, 1996.

(7) R. Bidou and F. Raynal. Covert Channels, http://www.radware.com/WorkArea/down-loadasset.aspx?id=3928. 2005.

(8) K. Borders and A. Prakash. Quantifying information leaks in outbound web traffic. In *2009 30th IEEE Symposium on Security and Privacy*, pages 129–140. IEEE, 2009.

(9) S. Bottcher, R. Hartel, and M. Kirschner. Detecting suspicious relational database queries. In *2008. ARES 08. Third International Conference on Availability, Reliability and Security*, pages 771–778. IEEE, 2008.

(10) R. Capizzi, A. Longo, VN Venkatakrishnan, and A.P. Sistla. Preventing Information Leaks through Shadow Executions. In *Computer Security Applications Conference, 2008. ACSAC 2008. Annual*, pages 322–331. IEEE, 2008.

(11) D. Caputo, M. Maloof, and G. Stephens. Detecting insider theft of trade secrets. *Security & Privacy, IEEE*, 7(6):14–21, 2009.

(12) National Computer Security Center. A guide to understanding covert channel analysis of trusted systems.

(13) President's Information Technology Advisory Committee. Cyber Security: A Crisis of Prioritization. 2005.

(14) DIANE Publishing Company. *National Computer Security Conference Proceedings, 1992: Information Systems Security*. DIANE Publishing Company, 1992.

(15) IBM Corporation. Protecting Sensitive Data in Non-Production Environments. 2008.

(16) D.E. Denning, P.J. Denning, and G.S. Graham. Selectively confined subsystems. In *Proc. International Workshop on Protection in Operating Systems. IRIA*, pages 55–61, 1974.

(17) Robert Drum. IDS and IPS placement for Network Protection. 2006.

(18) DLP Experts. The Evolution of Data Loss Prevention: Reducing Complexity (White Paper). 2010.

(19) F. Farrukh. Internet Architecture and Protocols; RAS, Firewall, IDS, IPS Comparison. 2007.

(20) Inc Fortinet. FortiOS Handbook v2: UTM Guide for FortiOS 4.0 MR2. 2010.

(21) Inc. Gartner. Magic Quadrant for Content-Aware Data Loss Prevention,. 2011.

(22) S. Gianvecchio and H. Wang. An entropy-based approach to detecting covert timing channels. *IEEE Transactions on Dependable and Secure Computing*, 2010.

(23) P.A. Gilbert and P. Bhattacharya. An approach towards anomaly based detection and profiling covert tcp/ip channels. In *Information, Communications and Signal Processing, 2009. ICICS 2009. 7th International Conference on*, pages 1–5. IEEE, 2009.

(24) V.D. Gligor and National Computer Security Center (US). *A guide to understanding covert channel analysis of trusted systems*. The Center, 1994.

(25) The Open Group. The What and Why of De-perimeterization. 2010.

(26) A. Hackl and B. Hauer. State of the Art in Network-Related Extrusion Prevention Systems. 2009.

(27) S. Harris. *CISSP certification all-in-one exam guide*. McGraw-Hill, Inc. New York, NY, USA, 2007.

(28) Forrest Heller. ICMPForge, http://www.forrestheller.com/ICMPForge/. 2009.

(29) V.C. Hu, D. Ferraiolo, D.R. Kuhn, Information Technology Laboratory (National Institute of Standards, and Technology). *Assessment of access control systems*. Citeseer, 2006.

(30) J.C. Huskamp. Covert communication channels in timesharing systems, Technical Report UCB-CS-78-02, Ph.D. Thesis, University of California, Berkeley, California. 1978.

(31) Infowatch. Global Data Leakage Report 2009. 2009.

(32) S. Katzenbeisser and F. Petitolas. Information Hiding Techniques for Steganography and Digital Watermaking. *EDPACS*, 28(6):1–2, 2000.

(33) R. Kemmerer. A practical approach to identifying storage and timing channels. 1982.

(34) R.A. Kemmerer. A practical approach to identifying storage and timing channels: Twenty years later. In *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*, pages 109–118. IEEE, 2003.

(35) S. Kuninobu, Y. Takata, H. Seki, and K. Inoue. An Information Flow Analysis of Programs based on a Lattice Model. *IEIC Technical Report (Institute of Electronics, Information and Communication Engineers)*, 100(472):25–32, 2000.

(36) B.W. Lampson. A note on the confinement problem. *Communications of the ACM*, 16(10):613–615, 1973.

(37) J. Marecki, M. Srivatsa, and P. Varakantham. A Decision Theoretic Approach to Data Leakage Prevention.

(38) F. Mayer, K. MacMillan, D. Caplan, and Safari Books Online (Firm). *SELinux by Example: Using Security Enhanced Linux*. Prentice Hall, 2006.

(39) S. McClure, J. Scambray, and G. Kurtz. *Hacking exposed 6: network security secrets & solutions*. McGraw-Hill Osborne Media, 2009.

(40) T.S. Messerges, E.A. Dabbish, and R.H. Sloan. Examining smart-card security under the threat of power analysis attacks. *IEEE Transactions on Computers*, pages 541–552, 2002.

(41) J.K. Millen. Covert channel capacity. In *Proc. IEEE Symposium on Security and Privacy*, pages 60–66, 1987.

(42) R. Mogull (Securiosis). DLP Content Discovery: Best Practices for Stored Data Discovery and Protection, whitepaper. 2008.

(43) R. Mogull (Securiosis). Understanding and Selecting a Data Loss Prevention Solution. 2008.

(44) Symantec N. Penso, Security Engineer Netcom Malam-Team. The Concept of DLP.

(45) Steffen Nissen. Fast Artificial Neural Network Library, http://leenissen.dk/fann.

(46) United States Department of Defense. Trusted Computer System Evaluation: The Orange Book. Publication DoD 5200.28-STD. Washington: GPO 1985.

(47) J. Oltsik. It's Time For A New Name for Data Loss Prevention (DLP): Myopic industry-driven title has become obsolete. 2010.

(48) M. Owens. A discussion of covert channels and steganography. *SANS Report, March*, 2002.

(49) P. Papadimitriou and H. Garcia-Molina. A Model for Data Leakage Detection. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*, pages 1307–1310. IEEE, 2009.

(50) P. Papadimitriou and H. Garcia-Molina. Data leakage detection. *IEEE Transactions on Knowledge and Data Engineering*, 2010.

(51) W.J. Prior. *Virtue and knowledge: an introduction to ancient Greek ethics*. Taylor & Francis, 1991.

(52) Linnemann J.J. Sauerwein, L.B. Personal Data Protection Act: Guidelines For Personal Data Processors. 2001.

(53) E.J. Sebes and M. Stamp. Solvable problems in enterprise digital rights management. *Information Management & Computer Security*, 15(1):33–45, 2007.

(54) G. Shah, A. Molina, and M. Blaze. Keyboards and covert channels. In *Proceedings of the 15th conference on USENIX Security Symposium-Volume 15*. USENIX Association, 2006.

(55) F. Shi, J. Li, J. Tang, G. Xie, and H. Li. Actively learning ontology matching via user interaction. *The Semantic Web-ISWC 2009*, pages 585–600, 2009.

(56) Paul Simmonds. Architectures for a Jericho Environment. 2005.

(57) A. Singh, O. Nordstrom, A.L.M. dos Santos, and C. Lu. Stateless Model for the Prevention of Malicious Tunnels.

(58) L. Spitzner. Honeypots: Catching the insider threat. In *Computer Security Applications Conference, 2003. Proceedings. 19th Annual*, pages 170–179. IEEE, 2005.

(59) Symantec. Symantec Data Loss Prevention 11 Technical Presentation.

(60) J.B. Taylor. When Insiders Attack: Covert Data Channels.

(61) J. Theyer. Covert Data Storage Channel Using IP Packet Headers. 2008.

(62) D.F. Thompson. Understanding financial conflicts of interest. *New England Journal of Medicine*, 329(8):573, 1993.

(63) N. Vachharajani, M.J. Bridges, J. Chang, R. Rangan, G. Ottoni, J.A. Blome, G.A. Reis, M. Vachharajani, and D.I. August. RIFLE: An architectural framework for user-centric information-flow security. In *Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture*, pages 243–254. IEEE Computer Society, 2004.

(64) ir.R.H. van Wanroij. Beveiligingsmodellen, MLS, Bell-LaPadula: hoe zat dat ook al weer? *Informatiebeveiliging*, November, 2004.

(65) J. Wiegley. Designing Trusted Operating Systems, Lecture notes 11, California State University, Northridge.

(66) R. Yokomori, F. Ohata, Y. Takata, H. Seki, and K. Inoue. Analysis and implementation method of program to detect inappropriate information leak. In *Quality Software, 2001. Proceedings. Second Asia-Pacific Conference on*, pages 5–12. IEEE, 2002.

(67) Y. Yu and T. Chiueh. Enterprise digital rights management: Solutions against information theft by insiders. *Research Proficiency Examination (RPE) report*, 2004.