

# Adversary Modelling

Ole Kasper Olsen



Master's Thesis  
Master of Science in Information Security  
30 ECTS  
Department of Computer Science and Media Technology  
Gjøvik University College, 2005



The MSc programme in Information Security is run in cooperation with the Royal Institute of Technology (KTH) in Stockholm.

Institutt for  
informatikk og medieteknikk  
Høgskolen i Gjøvik  
Postboks 191  
2802 Gjøvik

Department of Computer Science  
and Media Technology  
Gjøvik University College  
Box 191  
N-2802 Gjøvik  
Norway

## Abstract

Security measures are countermeasures against some sort of adversary, or adversaries, and as such are based on sets of assumptions made with regards to the adversary or adversaries by a system's designers.

However, often the assumptions designers of said countermeasures have introduced are hard to ascertain. Often, such information is scattered around in white papers and implementation notes, or even only implicitly stated or not at all.

This thesis introduces a novel framework for use in such situations where one requires—in a quick and efficient manner—to get an overview over which assumptions the designers of a system have made with regards to its adversaries. This may be invaluable to customers who wish to ascertain whether or not the adversaries protected against are sufficient in the actual operating scenario of the system.

The framework can also be used to help in the early design process of systems as a tool alongside such methodologies as threat modelling, as it easily highlights possible attack vectors.

The framework can be used to simplify the work of documenting and clarifying assumptions prior to and during security effectiveness analysis, and it is shown to work well on several different cases.



## Sammendrag

I beslutningsprosesser som omhandler informasjonssikkerhet gjøres det alltid antagelser med tanke på fienden eller fiendene et system er utsatt for, altså fiendemodellen systemet må operere under. Ofte er informasjon angående fiendemodeller vanskelig å finne. Denne oppgaven tar for seg å utvikle et rammeverk for lettere å kunne samle slik informasjon.

Rammeverket som foreslås vil være verdifullt i situasjoner hvor man ønsker på en rask og enkel måte å få oversikt over de antagelser som er gjort med tanke på fienden, for eksempel når man vurderer innkjøp av bedriftskritiske systemer. Man kan også lettere identifisere åpenbare mangler i fiendemodellen til systemet under vurdering.

Videre vil det presenterte rammeverket være til hjelp i systemutviklingsprosjekter som et hjelpemiddel for å tidlig fastslå hva slags fiender et system vil være utsatt for. Man kan da gjøre disse antagelsene på et tidlig tidspunkt, og være bevisst når det gjelder å dokumentere disse.



## Preface

You are now holding (or perhaps currently scrolling through) my master's thesis concluding my 2-year study in the field of information security. It has been a tremendously interesting study, with great fellow students and lecturers.

During the course of the six months assigned to the extended research project leading up to the final product in form of this master's thesis, I have learnt much about a vast array of different information security areas due to the nature of the thesis' topic.

The project has taken some twists and turns from what I initially envisioned. However, because of that, I feel that the thesis has become much more of a useful product, something for which I wholeheartedly thank my supervisor, Professor Einar Snekkenes. I would also like to thank Hanno Langweg at NISlab for valuable feedback and discussion.

Further, I extend a special thanks to fellow master students Fredrik, Anders, Ole Martin and Torkjel for invaluable feedback, numerous quarrels and squabbles, feasts, parties, concerts, rabid DooM II matches, fanatic cyberstalking, heated arguments and exhilarating moments of joy. You're the best.

Ole Kasper Olsen, June 30th, 2005





## Contents

<b>Abstract</b> . . . . .	<b>iii</b>
<b>Sammendrag</b> . . . . .	<b>v</b>
<b>Preface</b> . . . . .	<b>vii</b>
<b>Contents</b> . . . . .	<b>ix</b>
<b>List of Figures</b> . . . . .	<b>xi</b>
<b>1 Introduction</b> . . . . .	<b>1</b>
1.1 Topics Covered by This Thesis . . . . .	1
1.2 Problem Description and Research Questions . . . . .	1
1.3 Motivation . . . . .	1
1.4 Method . . . . .	2
1.5 Thesis Outline . . . . .	2
<b>2 Adversary Models</b> . . . . .	<b>3</b>
2.1 Introduction to Adversary Models . . . . .	3
2.2 Anonymity Networks and Services . . . . .	3
2.3 The Malicious Insider . . . . .	5
2.4 Cryptographic Algorithms . . . . .	6
2.4.1 Ciphertext-Only Attack . . . . .	10
2.4.2 Known Plaintext Attack . . . . .	10
2.4.3 Chosen Plaintext Attack . . . . .	10
2.4.4 Chosen Ciphertext Attack . . . . .	10
2.5 Cryptographic Protocol Analysis . . . . .	11
2.6 Discussion . . . . .	12
<b>3 Adversary Modelling</b> . . . . .	<b>13</b>
3.1 Threat Modelling . . . . .	13
3.2 Attack Modelling . . . . .	13
3.3 Protocol Analysis . . . . .	13
3.4 Discussion and Comparison . . . . .	14
<b>4 A Framework For Adversary Models</b> . . . . .	<b>17</b>
4.1 Principals . . . . .	17
4.2 Channels . . . . .	18
4.3 Protected Asset . . . . .	18
4.4 The Adversarial Setting . . . . .	19
4.4.1 Adversaries . . . . .	19
4.4.2 Intra-Adversary Channels . . . . .	21
4.5 Framework Summary and Modelling Pointers . . . . .	22
4.5.1 Summary of Notation . . . . .	22
4.5.2 Modelling Procedure . . . . .	22
4.5.3 Modelling Special Cases . . . . .	23
<b>5 Applying the Framework</b> . . . . .	<b>25</b>
5.1 Steganography . . . . .	25

5.1.1	Classic Steganography . . . . .	26
5.2	Digital Rights Management . . . . .	27
5.3	Onion Routing Networks . . . . .	29
5.4	Access Control/Local Attacks . . . . .	31
5.4.1	Symbolic link Attack . . . . .	31
5.5	Single Sign-On . . . . .	32
5.5.1	Kerberos . . . . .	33
5.5.2	Mobile Phone-Based Personal SSO System . . . . .	35
5.6	Database Security . . . . .	37
5.6.1	Statistical Attacks . . . . .	37
5.7	Wireless Networks . . . . .	38
5.7.1	The Wired Equivalent Privacy (WEP) Security Protocol . . . . .	38
5.7.2	WEP—Post-Break . . . . .	40
5.8	Authentication Systems . . . . .	41
5.8.1	Biometrics . . . . .	41
5.9	Web Application Security . . . . .	44
5.9.1	Simple Login Sites . . . . .	44
5.9.2	eCommerce . . . . .	45
5.9.3	Online Banking . . . . .	47
5.9.4	eVoting . . . . .	48
<b>6</b>	<b>Discussion . . . . .</b>	<b>53</b>
<b>7</b>	<b>Future Work . . . . .</b>	<b>55</b>
<b>8</b>	<b>Conclusions . . . . .</b>	<b>57</b>
	<b>Bibliography . . . . .</b>	<b>59</b>
<b>A</b>	<b>Paper Submitted for Publication . . . . .</b>	<b>69</b>

## List of Figures

2.1	Cryptography (encryption and decryption).	7
4.1	The Framework	17
4.2	Principals with a Connecting Channel	18
4.3	Principals and an Adversary	19
4.4	Principals and Two Operational Adversaries	20
4.5	An example of degrees of compromise in a principal	21
4.6	Intra Adversary Channels	22
4.7	Summary of Framework Notation.	23
4.8	Example of Modelling the Interaction Between a Process and the File system	23
5.1	The Steganographic Process.	26
5.2	Adversary Model of a Classic Steganography Scenario.	27
5.3	Digital Rights Management.	28
5.4	Adversary Model of a Digital Rights Management Scenario.	28
5.5	Anonymity network adversary model overview.	30
5.6	Adversary Model of a System Under an Application Vulnerability Attack	32
5.7	The Kerberos Authentication Scheme	34
5.8	The Adversary Model of the Kerberos Authentication Scheme	34
5.9	A Personal Single Sign-On System for Mobile Phones	36
5.10	Adversary Model of a Database System Under Statistical Attack	37
5.11	Alternate View of the Adversary Model of a Database System Under Statistical Attack	38
5.12	The Wired Equivalent Privacy Security Protocol	39
5.13	Adversary Model of a Wireless Network.	39
5.14	Adversary Model of a Wireless Network with WEP	40
5.15	Zero-Effort Biometric Authentication Adversary	42
5.16	Active Biometric Authentication Adversary	43
5.17	Adversary Model of a password-protected Web forum.	44
5.18	Forum Server under SQL Injection Attack.	45
5.19	Adversary Model of Electronic Commerce	46
5.20	Adversary Model of Online Banking with Mutual PKI-based Authentication	47
5.21	The Internet Voting System of Chen et al	50
5.22	Adversary Model of Chen et al's Voting Scheme Over the Internet	50
5.23	Extended Adversary Model of Chen et al's Voting Scheme Over the Internet	52



# 1 Introduction

*“[The Adversary:] An entity that attacks, or is a threat to, a system.” [107]*

## 1.1 Topics Covered by This Thesis

Without the adversary, there would be no field of information security; it is a defensive field of research and implementation.

This thesis revolves around the difficulty of easily seeing what kind of assumptions designers of security-related solutions and countermeasures have made with regards to the adversary or the adversaries a system faces.

This set of assumptions made with regards to the adversary are largely what constitutes an adversary model. This thesis will investigate whether there exists common adversary models within certain fields of information security, and whether good methodologies for assessing these adversary models exist.

We will also implement a framework for modelling adversaries. This framework will be put to the test on cases from such diverse areas of information security as database security, anonymity networks, steganography, single sign-on solutions, electronic commerce and banking, wireless networks and malware.

## 1.2 Problem Description and Research Questions

In many cases, it is hard to gauge the effectiveness of claimed countermeasures against information security related breaches without a clearly defined adversary model. Such much-needed information about adversaries are often either implicitly stated or scattered around in technical documentation. Having a method to quickly assess the adversary model used in such implementation will greatly improve an analyst’s ability to determine whether implemented countermeasures are sufficient.

Two research questions was defined;

- Can a simple modelling technique help ascertain assumptions made with regards to the adversaries in a system’s environment?
- Will the use of data flow diagram-like modelling techniques help the creative process of identifying a system’s potential adversaries?

## 1.3 Motivation

This thesis sets out to alleviate the problem of gathering scattered, inaccessible and implicit information about assumptions made with regards to the adversary. Succeeding in

this task will provide professional analysts and laymen alike a potent way to gather such information.

It is also a goal and motivation to provide a novel way of analysing a system from an information flow point of view, leading to a simpler recognition of weaknesses.

## **1.4 Method**

Initially, a thorough literature study of adversary models and methodologies for assessing adversaries or adversary models will be conducted. Existing methodologies will be qualitatively compared, highlighting the advantages of each methodology. As it is not expected that any current methodologies are suitable to gather assumptions made with regards to an adversary based on actual implemented countermeasures, a new framework will be developed to, in an efficient manner, get an overview of the adversary model or adversarial setting. The framework will then be applied to a number of heterogeneous cases to gauge its efficiency and expediency.

## **1.5 Thesis Outline**

We will begin in chapter 2 by introducing adversary models, and looking at some common adversary models in different areas of information security. Then, in chapter 3 we'll look at different forms of adversary modelling, or methodologies used to ascertain information about the adversary or adversaries a system faces.

In chapter 4, the framework for adversary modelling is introduced. The framework is founded on principles recognised in the previous two chapters. We then go on to show that the framework works on a diverse set of cases in chapter 5. Chapter 6 discusses the applicability of the framework based on findings in chapter 5.

## 2 Adversary Models

### 2.1 Introduction to Adversary Models

By the term “adversary model” we mean the set of assumptions, explicit and implicit, which have been made with regards to the adversary in any given situation.

While there is precedence for using such a definition of adversary modelling (e.g., [101, 128]), it is not widely used in literature. As mentioned previously, the adversary model is usually not even described in detail.

In the next sections, we’ll informally highlight prevalent adversary models within certain fields of information security.

### 2.2 Anonymity Networks and Services

Anonymity services deals with the problem of traceability on the Internet, which for the sake of privacy may not always be beneficial. On the field of anonymity services and protocols, David Chaum is a pioneering scientist, who in 1981 devised a technique based on public key cryptography that provided untraceability of email messages without involving a trusted third party [19]. The technique has later been dubbed “mix-nets”, and consists of several “mixes”. When a packet travels from the sender to the recipient, it is passed through a non-random, but difficult to predict, pattern of mixes, where each mix only knows the previous and next mix in the chain. This provides much better protection from straight-forward traffic analysis than for example anonymising proxy solutions, such as The Anonymizer<sup>1</sup>, where monitoring the proxy will make an adversary figure out who is communicating with whom.

Chaum’s original technique was developed with electronic mail in mind, which is a protocol with no real-time requirements. Thus, it is not directly applicable to low-latency protocols which require near real-time interaction, such as the HTTP protocol (users may expect an answer instantly). Onion Routing [47] is such a low-latency implementation of a variation of a mix-network anonymity system which creates a difficult to predict virtual circuit through an array of routers between sender and recipient which works with many different protocols, of which HTTP is an example. The established circuit will then transmit data in both directions, accommodating near real-time demands. The premise of anonymity is based on the fact that there are many users of the service, as onion routing and similar techniques are basically attempting to hide users among a crowd of other users. Recently, the Onion Routing initiative’s “Tor” [35]—a second generation onion router was announced. “Tor” is, even though certain attack vectors have been identified [82], believed to be a adequately good solution and is supported

---

<sup>1</sup><http://www.anonymizer.com> (last visited June 30th, 2005)

by the Electronic Frontier Foundation<sup>2</sup>. Other implementations of similar techniques for low-latency use include Crowds [93], Freedom [12] (discontinued commercial implementation), Web MIXes [8] and Rennhard et al's implementation [96]. There are also several implementations for use with services where there are no real-time response requirement such as email (e.g. Babel [51], Mixmaster [79] and Mixminion [27]), and also systems which provide ephemeral anonymous peer-to-peer connections (e.g. Tarzan [44] and MorphMix [95]).

Chaum also devised another anonymity system, the "DC-net", garnering its name from the solution to the Dining Cryptographers problem [20]. DC-nets can be shown to be guaranteed anonymous by way of information theory, however they are not considered feasible to implement in large, distributed networks. Therefore, mix-nets remains the solution with the most promise for widespread use.

The adversary models present to a mix-net may be many. Assessing the security of "Tor" [118], Syverson et al define the following adversary models:

**Observer** The Observer is the typical eavesdropping adversary. He may monitor, but not initiate connections.

**Disruptor** A disrupting adversary is an adversary with the ability to delay the traffic to and from a link in an onion routing network. By delaying the traffic into a link, he may be able to see disruptions in the delivery of packages in a different part of the network.

**Hostile User** The hostile user is a legitimate user of the network and may initiate and destroy connections using specific routes through the onion routing network.

**Compromised COR** A compromised COR (Core Onion Router, a mix in the onion routing network model) is the strongest type of adversary considered by Syverson et al. A core onion router being an integral part of the onion routing network this adversary can manipulate any connection that it controls and also create new connections that pass through itself.

In the assessment of "Tor's" security, Syverson et al conclude that it is sufficient to assess security in light of a compromised COR, as the other defined adversaries may only perform subsets of the compromised COR's possible actions.

The aforementioned adversaries are basic types of adversaries, and compounded adversaries (or cooperating adversaries) may be comprised of any number and combinations of basic adversaries. The number of cooperating adversaries is a significant point regarding anonymity networks, and Syverson et al describe the following compositions in view of compromised COR adversaries which they consider the strongest:

**Single Adversary** A lone adversary.

**Multiple Adversary** A fixed number of randomly distributed subset of compromised CORs.

**Roving Adversary** Related to the multiple adversary model, however the compromised CORs may change at specific intervals.

---

<sup>2</sup><http://tor.eff.org> (visited June 30th, 2005)



**Global Adversary** All CORs are compromised.

The global adversary is a formidable one, and one which Onion Routing and other mix-net approaches provide no protection against. If all mixes are compromised, this compound adversary will at any time have full control over who is talking to whom on the network.

In other literature where a more general approach to mix-nets are taken, e.g. [94], many of the same adversaries as presented by Syverson et al are echoed. In [94] a distinction between internal and external adversaries are made. An internal adversary is one who controls a mix on the network and the external is simply an observing adversary. These adversaries are equal to Syverson et al's *compromised COR* and *observer* adversaries. There are also references to partial and global adversaries which echoes Syverson et al's compounded adversaries, in addition to the distinction between an active and passive adversary. The passive adversary is an *observer*, while the active adversary may be any one of the other above-mentioned.

### 2.3 The Malicious Insider

Within any corporation, one of the most severe and expensive security risks is having a malicious insider among the staff.

Bradley Wood has developed an adversary model [128] of the malicious insider for use in red team (penetration testing) attacks on corporate computers systems. In his model, Wood describes an adversary who have unrestricted access to (parts of) key infrastructure within the company, and the skills and knowledge to exploit his access—his knowledge regarding the target system will be extensive. Often, the malicious insider is a privileged user on the system in question, and thus may not even have to escalate his current privileges in order to mount an attack.

Wood describes the insider as a very risk averse adversary, given that discovery would be the ultimate defeat. He has no way to run nor hide. For this reason, the adversary generally works alone, and will only rely on colleagues when absolutely necessary. The adversary may use social engineering to rely on a colleague doing his bidding without even knowing the malicious intent.

Wood's insider is often motivated by profit or a personal motive, such as revenge, or other consequences which is undesirable for the company he works in. Wood reckons this type of adversary being a person with a character defect, or an operative from a competitor.

Wood is not alone in focusing on the malicious insider, however. As we move into a more and more digitalised society it has become clear for an increasingly larger group of people that the unprecedented amount of trust the developers and operators of large computer systems were given opens up for a host of severely dangerous situations. Many of the aspects of the malicious insider seen from a more psychological point of view are addressed by Shaw et al in [106]. They also make some assumptions which Wood does not explicitly make, e.g., that the insider adversary is one whose job is concerned with the information systems of the company. He is not merely an end user who uses a computer as a tool. The general tone of Wood's adversary supports this notion, although not stated explicitly.

There has also been conducted thorough research on malicious insiders jointly by the United States Secret Service and Carnegie Mellon's CERT [91, 59].

A recently published report [77] by Deloitte based on a survey among banks and other financial businesses showed that the insider threat has increased in later years. In 2004, 14% of the responders had encountered attacks from an inside source, a number which rose to 35% in the 2005 survey.

Wood's adversary has capabilities that highly depends on his skills. At the very least, this adversary is able to gather intelligence without arising any suspicion, based on his familiarity with the target. The adversary *may* be a local domain expert on the target, meaning he *may* do anything within the capabilities of his tools. What the adversary is capable of may also be fuelled by his motive, which may be profit, change-provocation, company subversion or some other personal motive (i.e., hate and revenge). Wood details a lot of potential capabilities the adversary may have.

The insider is assumed to have access to most resources within the target system (coworkers, software and hardware), but not necessarily all. As for computational resources, he will have access to most of the company's computers and other hardware.

The insider is by his own a very capable adversary, however he may not have direct access to all necessary systems. To access a certain system, he may try by social engineering to fool someone, or he will need to compel colleagues into working with him as an accomplice. He will however, do this only when it is absolutely necessary, as he is very risk averse. So, usually the adversary will work alone.

One very likely accomplice however, is the external employer who may have paid the insider to attack the target.

## 2.4 Cryptographic Algorithms

The reasoning behind encryption algorithms are largely based on the work of two persons. Auguste Kerckhoffs' principles written for the French military [61], dating back to 1883, and Claude Shannon's communication theory for what he calls "secrecy systems" [105], published in 1949 following his seminal paper which single handily introduced the world to the science of information theory [104].

The key element of Kerckhoffs' work is his six principles for cipher systems<sup>3</sup>:

1. The system must be practically, if not mathematically, indecipherable;
2. It must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience;
3. Its key must be communicable and retainable without the help of written notes, and changeable or modifiable at the will of the correspondents;
4. It must be applicable to telegraphic correspondence;
5. It must be portable, and its usage and function must not require the concurrence of several people;
6. It is necessary, given the circumstances that command its application, that the system be easy to use, requiring neither mental strain nor the knowledge of a long series of rules to observe.

These outline prudent rules to follow when designing a cryptosystem. Even though these principles were written more than 100 years ago, they are still as important today,

---

<sup>3</sup>English translation from Wikipedia [http://en.wikipedia.org/wiki/Kerckhoffs'\\_law](http://en.wikipedia.org/wiki/Kerckhoffs'_law) (last visited June 30th, 2005)

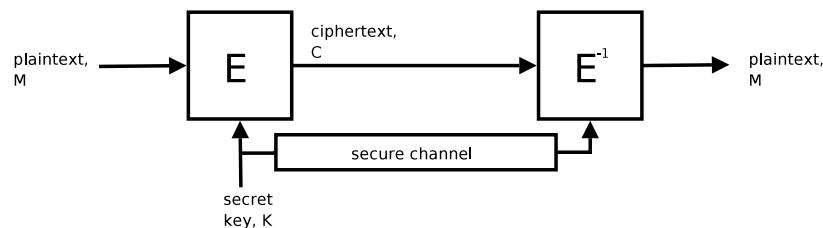


Figure 2.1: Cryptography (encryption and decryption).

some more than others. Kerckhoffs' perhaps most important principle (number 2) states that the security of a cryptosystem should only be dependent of the key used when encrypting and decrypting. In other words, if any part of the cryptosystem should fall into an adversary's hands, the adversary would gain no advantages with regards to cryptanalysis. In addition there would be no logistical inconveniences regarding replacing the entire algorithm in the case of compromise, as one may merely change the key or keys, rather than inventing a new cryptosystem.

A cryptosystem is comprised of a set of primitives. It has a finite set of messages, or plaintexts,  $\mathcal{M}$ . These plaintexts map to a finite set of ciphertexts,  $\mathcal{C}$ . To transform a plaintext into a ciphertext, an encryption function,  $e_K \in \mathcal{E}$  exists, where  $\mathcal{E}$  is the set of encryption functions each taking a key  $k \in \mathcal{K}$  as a parameter, where  $\mathcal{K}$  is the keyspace, which is a finite set of possible keys. Transforming the ciphertext back to plaintext is done by using a decryption function,  $d_{k'} \in \mathcal{D}$ . In symmetric cryptosystems the encryption and decryption keys ( $k$  and  $k'$ ) are equal. This is schematically shown in figure 2.1.

Given these definitions, one can say that Kerckhoffs' principle makes the assumption that an adversary knows at least  $\mathcal{M}$ ,  $\mathcal{C}$ ,  $\mathcal{K}$ ,  $\mathcal{E}$  and  $\mathcal{D}$ . The adversary is not assumed to know the particular key or keys used,  $k$  and  $k'$ , hence the security of the system relies on the secrecy of these keys.

Most of today's important symmetric and asymmetric cryptosystems are designed in such a way that the inner workings may be published without inconveniencing the user of the cryptosystem (e.g., DES [84], AES [24, 85]), and following the invention of public key cryptography by Diffie and Hellman [34], cryptographic algorithms exist that rely on public exposure (e.g., RSA [97], ElGamal [40]). These cryptographic algorithms adhere to Kerckhoffs' principle by placing the burden of secrecy on the usage of one or several keys or keystreams. In other words, the adversary is assumed to be in possession of detailed information about the inner workings of the cryptosystem. Any other approach is often considered to be naive. In 1994, RSA Laboratories' RC4 stream cipher, developed by Ronald Rivest, was reverse engineered and made publicly available on a cryptography related Internet mailing list<sup>4</sup>. Even though RSA are still only revealing the RC4 design to licensees, there are now "RC4 compatible" stream ciphers in widespread use.

When the strength or secrecy level of cryptographic algorithms is assessed, a computational view of the adversary is used. Certain assumptions about the processing capabilities of the adversary are made, and the level of secrecy is based on those assumptions. As a basis for most of the computational models, lies Shannon's theories of secrecy in communication. Shannon's seminal paper of 1949 [105] defines the theory of what the level

<sup>4</sup>RC4 appeared on the "Cypherpunks" mailing list in September of 1994.

of secrecy in a cryptographic system might be, and his teachings are still as important today as when they were published in 1949.

Shannon divides the field of information secrecy into two distinct areas: *theoretical secrecy* and *practical secrecy*.

Within the field of theoretical security, Shannon assumes an adversary with unlimited time and resources. In theoretical secrecy, there is only one secure cryptosystem; the one-time pad. Shannon proves that a one-time pad is unbreakable—it has “perfect secrecy”. Shannon’s proof is based on the fact that if a cryptosystem were to have perfect secrecy, the adversary would not be able to gain any knowledge about the plaintext after intercepting a ciphertext. In other words, the probability of deducing what the message might be, will be the same both before and after an encrypted version of the message is intercepted. This is only achieved when there are as many possible ciphertexts of a message as there are plaintexts, and every ciphertext is equally probable, that is  $P(c|m) = P(c)$ —the probability of a certain ciphertext is independent of the message  $m$ . This makes successful statistical analysis of the ciphertext impossible, because the ciphertext retains absolutely none of the characteristics of the original language.

The Vernam cipher [124] has this property<sup>5</sup>; it uses an XOR operation on one letter of the alphabet of plaintexts and one letter of randomly generated key of the alphabet of keys. Thus, given an example alphabet of  $\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0, 1\}$  we have that  $P(0 \rightarrow 1) = P(0 \rightarrow 0) = P(1 \rightarrow 0) = P(1 \rightarrow 1) = 0.5$ . As Schneier points out in [100], the one-time pad is largely useless in today’s applications because the key is as large as the plaintext. We are then stuck with a key distribution problem, rather than a plaintext distribution problem, and in the vast majority of cases, it is equally difficult to securely distribute the key as the plaintext. However, it is important in a theoretical context to be aware of the fact that with any cryptosystem which uses a smaller key than plaintext, cryptanalysis is possible, given an adversary with enough time and resources.

In the field of practical secrecy, where a one-time pad might not be practicable, Shannon describes how secure a cryptosystem is against cryptanalysis by using the average amount work needed to determine the key as a metric. In Shannon’s time, man hours may have been the best measure, today this ties into the computational resources of the adversary. This notion of using amount of work as a metric, is echoed in other areas of information security as well. As an example, we have Schudel and Wood’s paper on adversary work factor within the area of information assurance [102].

While Kerckhoffs and Shannon lay down many of the important aspects of a cryptosystem adversary, one<sup>6</sup> of the most prolific methods of assessing security of cryptographic algorithms today is by way of computational complexity theory and analysis, even though also Shannon’s theoretical approach has been adapted to a modern setting by the likes of Martin Hellmann [52]. Hellmann argues that computational complexity theoretical analysis will be influential on the practise of cryptography, while classical theory (Shannon theory) will provide valuable insights into important design principles. The theory of Shannon from [104], also finds its way into computational complexity based theory and application [129].

To implement computational complexity theory in a practical way, so-called “notions” are commonly used. The concept was first introduced by Goldwasser and Micali in [48],

---

<sup>5</sup>Shannon proved that any unbreakable cipher would be homologous to the Vernam cipher [105].

<sup>6</sup>A complementary method is that of computability rooted in Turing’s papers [109, 121].

and is a notion of certain attributes and properties a perceived secure algorithms should have. The security may then be judged by how probable it is that the notion holds. As an example of concrete implementation of computational complexity-based security assessments of a chosen plaintext adversary (see section 2.4.3), we have Bellare et al's study of symmetric encryption algorithms [7]. They give the following four notions for secure symmetric encryption systems, of which “find-then-guess” and “semantic” security echoes Goldwasser and Micali's notions, although taken from the public key context of Goldwasser and Micali's research to a symmetric key one.

**Real-or-Random** This describes the notion of an adversary which cannot distinguish between an encrypted plaintext and an encrypted randomly generated string, or—in other words—the adversary is not able to distinguish between the encryptions of two equal-length strings. In Bellare et al's chosen plaintext setting, the adversary has the advantage of being able to query an oracle which may or may not answer with the encrypted version of the plaintext the adversary provided.

**Left-or-Right** This notion is related to the first one described. The adversary will query the oracle for an encryption of two equal-length strings. The oracle will answer with an encrypted version of one of the strings. Given a good cryptosystem, the adversary must not be able to deduce which of the strings he provided was encrypted.

**Find-then-Guess** The Find-then-Guess notion describes an adversary who actively finds two plaintexts he want encrypted. After having saved some state information regarding these strings he submits them to the oracle. The oracle will then replay with then encrypted version of one of the strings. Given a good encryption system, the adversary will only be able to guess correctly 50% of the times, with only statistically insignificant deviations.

**Semantic** The notion of semantic security dictates that whatever may be efficiently computed about the plaintext by the adversary given a ciphertext, may also be computed without the ciphertext. In other words, there should be no advantage in knowing the ciphertext of a plaintext. Note that this is Shannon's definition of a “perfect secrecy” system, adapted to a computational complexity setting (restricted to adversaries with polynomially bounded resources available) for symmetric encryption.

In [37], the authors introduce a cryptosystem which remains non-malleable even under an adaptively chosen ciphertext attack (see section 2.4.4).

**Non-Malleability** First introduced by Dolev et al [37] as an extension to the notion of semantic security, non-malleable cryptography ensured that it would be impossible to generate a different ciphertext so that the respective plaintexts are related.

Traditionally, a cryptosystem has been considered broken if the notion of semantic security is violated. In [7] it is shown that the real-or-random notion is equivalent to the left-or-right notion (there are security preserving reductions between them). They also show that the notion of semantic security is equivalent to the notion of find-then-guess. The reduction between left-or-right indistinguishability to find-then-guess security is security preserving, however the inverse is not. This means that to prove the security of a

cryptographic system, one should prove real-or-random or left-or-right indistinguishability, as that will imply good reductions to the other notions.

The cryptographic community usually describe four main models of the adversary. It must be assumed that all of the following classes of adversaries know the details of the cryptosystem in use. In each of the adversary models, the ultimate goal of the adversary is to gain knowledge of the key used in the cryptosystem, alternatively find a general algorithm which can transform the ciphertext into plaintext under the current key more efficiently than brute force attacking the key.

#### **2.4.1 Ciphertext-Only Attack**

This adversary has access to the ciphertext of a message. In other words he may be eavesdropping on the communication channel and may intercept messages as they travel across. In modern-day cryptography, successfully carrying out a ciphertext-only attack is difficult, due to the extreme complexity of the cryptographic algorithms in wide use today.

#### **2.4.2 Known Plaintext Attack**

In addition to the ciphertext, this adversary also has access to the plaintext equivalent of a ciphertext. The adversary may have obtained the plaintext-ciphertext message pair by anticipating the contents of the message, or via other channels. He has, however, not generated the pair himself. By knowing one or several plaintext-ciphertext pairs, the adversary will be able to simplify the cryptanalysis process.

#### **2.4.3 Chosen Plaintext Attack**

The adversary has the ability to choose a plaintext and observe the resulting ciphertext. In other words, the adversary has, although often temporary, access to an “encryption device”—often called an *oracle*. Given the temporary nature of the oracle, the chosen plaintext attack is often called the “lunchtime attack”, referring to the adversary sneaking in and using the encryption device while the office staff is out for lunch. The fact that this adversary has the ability to choose plaintexts to be encrypted, does not mean that he knows the key, but is exploiting an existing encryption mechanism which uses the correct key.

The chosen ciphertext adversary is the weakest adversary when dealing with public-key cryptography, as he will always be able to encrypt a plaintext of his own choice.

##### **2.4.3.1 Adaptive Chosen Plaintext Attack**

This adversary is a variation of the chosen plaintext adversary. The important difference is that here, the adversary based on the resulting ciphertexts may adapt the new plaintexts given to the encryption device so that the cryptanalytic gain from new ciphertexts will be higher.

#### **2.4.4 Chosen Ciphertext Attack**

The chosen ciphertext adversary is an adversary model first formalised by Naor and Yung [83], who has the ability to probe a “decryption device” polynomially many times with ciphertexts of his own choice. The decryption device, or decryption oracle, will provide the corresponding plaintexts of the given ciphertexts. He will then attempt to break the system without the aid of the oracle. That is, formally the decryption oracle will be unavailable at the time the adversary is about to break a challenge ciphertext. The same

applies for this adversary as for the chosen plaintext adversary, in that he does not know the key, but is exploiting an existing decryption mechanism which uses the correct key. This is considered the most powerful adversary model, as the adversary has access to a decryption mechanism which can decipher any messages given to it, provided the key does not change. The fact that the adversary has the capability of deciphering any message, is in itself an obvious security malfunction, however it is often assumed that the adversary will only have access to the decryption device for limited time spans, as is the case of the chosen plaintext attack.

An example of a chosen ciphertext attack is an attack on the WEP protocol for security in wireless networks, first shown by Borisov et al [11], where a wireless access point is fooled into decrypting messages of the adversary's choice.

#### 2.4.4.1 Adaptive Chosen Ciphertext Attack

In [90], Rackoff and Simon formalises a stronger adversary than the one Naor and Yung considered. Rackoff and Simon's adversary has access to the decryption oracle even after the challenge ciphertext which the adversary is about to break has been given to him. He may not query the oracle for a decryption of the challenge itself, but any other ciphertext is acceptable. He may then adapt the queries to the oracle based on the given challenge.

Common in-use cryptographic algorithms, such as RSA, are not secure against a chosen ciphertext adversary. Cramer and Shoup developed the first efficient public key cryptosystem [23] which is secure against a chosen ciphertext adversary, which is based on the "hard" problem (and therefore "provably secure" (see [114] for an example)) Decision Diffie-Hellmann [10].

## 2.5 Cryptographic Protocol Analysis

In literature regarding cryptographic protocol analysis, there are generally two directions regarding adversary models. The perhaps most well-known and most widely used approach to protocol analysis is the *formal* approach, where the cryptographic operations of the protocol may be seen as formal expressions. This approach was largely introduced by Dolev and Yao in the seminal work "On the Security of Public-Key Protocols" [38].

The other approach is the *computational* approach to protocol analysis. This approach takes the analysis philosophy of cryptographic algorithms and applies it to cryptographic protocols. Previously, this has been considered too laborious and difficult to apply to entire cryptographic protocols.

As a side note, it is usual to make the distinction between passive and active attacks on protocols, however, assuming only a passive attacker who may only eavesdrop messages is considered naive.

The formal approach was, as mentioned, in many ways initiated by the publication of [38], of which an extended abstract appeared in 1981 in the proceedings of IEEE's 22nd annual symposium on Foundations of Computer Science. However, as early as in 1978, Needham and Schroeder had already hinted at the same [86]. Although not as formal in their proofs of protocol security, they introduced some of the same characteristics of an adversary as Dolev and Yao.

The formal methods operate under a simplifying set of assumptions. Without these simplifications, one cannot guarantee that an adversary may guess e.g. keys without using computational complexity theory as is prevalent in the assessment of the security

of encryption algorithms.

The main characteristic coined by Needham and Schroeder is that the adversary is able to connect a computer to all communication paths, thus being able to manipulate transmitted messages in many different ways. The most important aspects of this is that the adversary must be expected to be able to alter parts of messages, replay messages and send false messages. Dolev and Yao label their adversary an “*active*” *eavesdropper* as he will first eavesdrop on the communication channel, then attempt anything within his power to decrypt the messages. They assume that the adversary is a legitimate user of the network, thus he may legitimately send messages.

As Meadows points out in [75], the Dolev-Yao adversary will always know which individual parts a message consists of. I.e., a message containing  $E_k(x)$  is known by the adversary to contain the message  $x$ , encrypted with an encryption algorithm taking a key,  $E_k$ . In other words, it is assumed that the messages travelling over the network is not merely seen as an incomprehensible string of binary digits by the adversary, but as the individual components and primitives the message consists of. Also, the adversary is restricted to the same set of operations as the principals involved in the protocol [1].

One very important assumption made with regards to the adversary in this model, is that the cryptographic algorithms are considered secure and for all intents and purposes, unbreakable. In other words, the cryptographic operations of encryption and decryption are considered primitives, and the adversary is assumed to not be able to do anything to compromise the security by means of breaking the encryption.

The computational approach makes roughly the same assumptions about the adversary as explained in the previous section on encryption algorithms.

## 2.6 Discussion

The adversary models shown above make the foundation of the framework for adversary models presented later in the thesis.

The assumptions made with regards to the adversaries in the previous sections in this chapter, show that focus needs to be on three main areas with regards to the adversaries. First among these are the adversaries’ capabilities, as in logical operations they can calculate. Then there is the amount of resources the adversary has available. These resources may be of any type, such as computational or monetary, although monetary resources may easily be reduced to other forms of resources. Thirdly, the access to communication channels or channels where information flows is vital, along with the set of operations the adversary may do towards these channels.

Also, the study of anonymity services shows that the extent of which several adversaries are able to cooperate is vital.



## 3 Adversary Modelling

There are different types of methodologies for modelling a system with regards to its security and to improve security and knowledge regarding security related decisions.

### 3.1 Threat Modelling

Threat modelling [116] is much used within software engineering and system development, i.e., in the initial phases of a product's or system's life cycle. Threat modelling is based on understanding the goals an adversary may have for attacking a system by focusing on the assets of the system. Threat modelling, as a part of a software engineering discipline, may produce vast amounts of documentation, something which is invaluable when implementing the system.

Threat modelling is often a very “heavy” phase during design of a system. As such, threat modelling may use many different notations to visually modelling the system. These include data flow diagrams, flow charts and UML.

CORAS [43] is a framework for threat modelling, which is wholly based on UML, and aims to provide a consistent set of documentation of a system's threats and security considerations.

Other frameworks include the very formal, documentation-oriented Common Criteria [120]. It is a very thorough framework, and aims to create a set of common criteria for evaluating security levels within systems and its origins may be found in the “Orange Book” [123], the US Department of Defense evaluation criteria from 1985. The existence of sets of common criteria is a key aspect of being able to trust and compare products which has received some form of security certification. As such, the Common Criteria is mostly used when the goal is to formally certify a product.

### 3.2 Attack Modelling

Attack modelling [74, 99, 112], unlike threat modelling, attempts to identify an adversary's full attack path into a system as smaller individual attack goals, and as such is more used within penetration testing to assess a system's level of security towards the end of the implementation phase, rather than in its design and conception phase.

Attack modelling do however focus clearly on the adversaries and his abilities to break a system via specified attack paths. In other words, much focus is put on the capabilities and resources of the adversary, more so than in most threat modelling methodologies.

### 3.3 Protocol Analysis

While protocol analysis often does not consider entire systems, but the communication between principals using the system, it still provides valuable insight into protecting a

system from adversaries, although at a different level of abstraction. This difference in abstraction may come into play in situations where a system may look perfectly fine in a threat model, but fail due to specific communication issues in a protocol analysis.

The modelling framework to be presented later also focuses on communication and information flow, however at an abstraction level closer to that of threat and attack modelling.

Following the publication of Dolev and Yao’s seminal work, many other formal methods for protocol analysis have been developed (e.g., [2, 15, 31, 60, 75]), however the assumptions made about the adversary are in many ways not dissimilar from the assumptions made in the case of the Dolev-Yao adversary as seen in section 2.5, even though the individual approaches to formal analysis may differ significantly.

Based on the nature of formal protocol analysis, the assumptions made are often implicit, and in some cases not even an act of conscious thought. Abadi and Rogaway argue that connections between the computational and formal approaches of cryptographic analysis should confirm or improve the relevance of formal proofs of protocol security [3] as implicit assumptions and gaps would become more obvious.

One of the most notable weaknesses of the formal approach to protocol analysis, is that it cannot sufficiently be used to explicitly prove that a certain protocol is secure except against attacks which are modelled into the analysis methodology. It is very good at highlighting possible attacks on a protocol, but it cannot explicitly prove that a protocol is secure against all adversaries having a specific amount of computational resources.

The computational approach to protocol analysis [2, 68] aims to remedy this by introducing some of the principles from assessing security of cryptographic algorithms. Some work have currently been done with regards to marrying the two approaches (e.g., [3, 89, 132]).

As evidenced by the previous chapter, the computational complexity approach to protocol analysis itself provides quite a set of assumptions about the adversary, even though not as strict as the formal analysis methodologies. Besides, as evidenced by [3], the computational complexity approach is often best used in combination with the power of the formal protocol analysis approach.

The formal approach to protocol analysis is by far the most widespread, even though some issues may still be unresolved [76].

### **3.4 Discussion and Comparison**

Of the aforementioned methodologies only the protocol analysis methodologies are assumed to be viably applied to already existing designs to judge security. However, as we in this thesis are focusing on adversary models—that is, the assumptions made with regards to an adversary or adversaries—the stringent assumptions introduced by especially formal protocol analysis methodologies may skew the true adversarial setting of a system.

The attack modelling and threat modelling approaches are more of an important discipline during the design stages of a system than a good way to evaluate an existing system’s resilience towards adversaries by a third party.

Our framework for adversary models which will be introduced in the next chapter aim to be a viable method to determine the suitability and strength of security features of a system by a third party, such as a customer. As such, our framework is based around the

concretely implemented security features of a system, to decide what kind of assumptions its designers have made with regards to adversaries.

The framework is light-weight and the process of modelling a system is thought to be fairly rapid. Knowledge about the system is required however, to identify the paths of communication and current security countermeasure implementations.

Our channel-based approach may also be valuable as a means to visualise adversaries previously not thought of. We will later show how our framework may be used to identify adversaries which are both protected against and not.



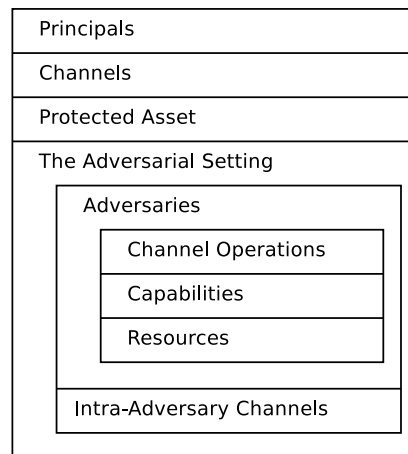
## 4 A Framework For Adversary Models

As described in previous sections, the adversary model of a certain system may not be explicitly stated, and clues to the adversary envisioned by the designer might be scattered around in the design and documentation of a system.

The aim of the following framework is to help focus on what we believe to be the most important properties of the adversarial setting a system faces. It will help to easier identify the underlying assumptions made with regards to the adversary or adversaries.

It will provide a convenient way to get an overview of a potentially chaotic area of inherent knowledge, at a convenient abstraction level.

The framework consists of the following key properties:



**Figure 4.1:** The Framework

### 4.1 Principals

Any adversary model will refer to a certain set of principal participants in a security system. The principals are the persons, computers or conceptual processes interacting within the system faced by a possible attack from an adversary or adversaries. Generally, this means any individual or system/machine with which the adversary may interact in any way.

Unlike some other methods, such as the Dolev-Yao approach to protocol analysis, we do not consider the adversary one of the principals.

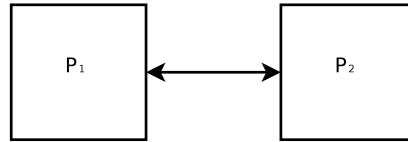


Figure 4.2: Principals with a Connecting Channel

## 4.2 Channels

The channels are what facilitate information flow between principals. Hence, the principals of the adversary model will interact through these channels. The channel over which they interact may be very different, depending on the adversary models and situations in which they apply, however there needs to be a connection between any two or more intercommunicating parties.

The different kinds of channels are characterised by the nature of the channel, the channel's direction and the channel's bandwidth.

Some examples of the nature of channels may be speech and vocal interaction, a physical cable in a network (where a network can be seen as multiple channels between several interconnected principals), sign language or other visual forms of communications and of course various forms of written communication. At a different level we also have the communication channels which are opened on top of underlying channels. On top of a network we may have specific channels defined by their operating protocol, etc—each potentially with their own set of interconnected principals.

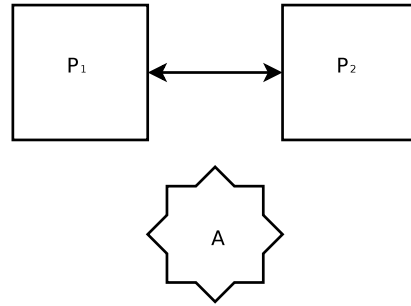
The direction of the channel is given by the direction information may flow. An example of this distinction may be found in high-security systems, such as those following the Bell-LaPadula model [6]. Here, information may only flow from a lower or equal level of security, often summarised as “no read up” and “no write down”. As a result a channel connecting a low-level security principal with a high-security one will only be one-directional; from the low-level principal to the high-level one.

The bandwidth of a channel may play a role in the modelling of the adversary, although often the channel has for all intents and purposes unlimited bandwidth. Limited bandwidth channels are for example often seen within the field of steganography or in other cases where forms of covert channels are in use.

The modelling of the channels is very reminiscent of Yourdon's well-known data flow diagrams [130], and catches the same notion of information flow. To illustrate principals and channels, see figure 4.2. It shows two principals,  $P_1$  and  $P_2$ , with a bidirectional channel between them for information exchange. This model notation will be used throughout the thesis; the principals are squares, and channels are solid lines.

## 4.3 Protected Asset

Information security is the defensive discipline of computer science, as security systems are being put in place to protect some asset. The goal of the adversary in a specified model will be to break the protection around this asset in order to obtain access to it. This asset may be something wildly different depending entirely on the adversary model and scenario. Examples includes anything from confidentiality and integrity of messages and anonymity of communicating parties to the protection of a person or a somehow valuable physical object.



**Figure 4.3:** Principals and an Adversary

We use here a similar definition of the protected asset as the international standard for security management, ISO/IEC 17799 [55]. That is, the protected asset, which is usually information of some form, has a certain value to the owner. A security-related breach in the countermeasures around the asset will lead to its value diminishing. E.g., compromise of confidentiality (the adversary now has the same information), integrity (the adversary has tainted the information) or availability (the adversary blocks timely access to the information) will diminish the value of the asset in the eyes of the owner.

## 4.4 The Adversarial Setting

We have up until now presented the necessary context in which an adversary may operate. The following sections will focus on the assumptions made with regards to the adversarial setting in which the system operate, given the assumptions made with regards to the context, as described in previous sections.

The adversarial setting comprises one or several adversaries, the channels which connects them, and the adversaries' interface with the system under attack. The individual adversaries' access to the different channels between the principals are defined by a set of possible channel operators.

### 4.4.1 Adversaries

An adversarial setting may consist of several adversaries. Each of these adversaries have their own set of operations they may perform on the channels between principals, and their own set of capabilities and resources.

To figuratively model an adversary, an eight pointed star is used, as in figure 4.3.

#### 4.4.1.1 Channel Operations

For each identified channel between principals, the adversary will have a certain set of operations which he may carry out on the channel, or on the messages which are transmitted over the channel.

The following operations are defined:

**Read** By having *read* access to a channel, an adversary will be able to monitor and read all messages which are being transmitted on the channel without exceptions.

**Intercept** The *intercept* operation is defined as the act of being able to block the transmission of a selected message over the channel.

**Write** With full *write* access to the channel, an adversary may introduce his own mes-

sages onto channel, meaning he may also replay messages he has *intercepted* or *read*. Often, full-fledged write access is only provided if the adversary is acting as a principal.

Different types of adversaries may be constructed from these basic channel operations. Syverson et al introduce some basic types of adversaries for use in their assessment of security in the Onion Routing anonymity network [118]. These include the “observer”, the “disruptor” and the “compromised COR” (COR is a router in the anonymity network). The “observer” is essentially an adversary with only *read* access one or more channels. The “disruptor” on the other hand has in addition to *read* access, also the ability to *intercept* messages and also *write* new content to the channel. The “compromised COR” is a compromised principal of the anonymity network, and has at least *read* and *write* access to its adjacent channels.

Figure 4.4 shows our system with two adversaries;  $A_1$  who can *read* the channel between the principals, and  $A_2$  who are able to do anything to the channel (i.e., it is an adversary with *read*, *write* and *intercept* operations). The dotted lines indicate the operations. Arrow pointing at the adversary indicates *read* operations, while an arrow pointing at the channel indicates *write* operations. A small box on the channel indicates the *intercept* operation.

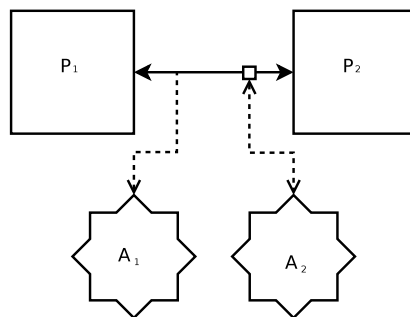


Figure 4.4: Principals and Two Operational Adversaries

To model degrees of compromise within a principal of a system, we can focus on the channels, and the principal’s adversarial element’s access to these at a different level of abstraction. As an example, the Machiavellian adversary [117], is a compromised principal which is more reluctant to part with secrets such as encryption keys, as opposed to the Dolev-Yao adversary [38] which is assumed to be a completely compromised principal. Figure 4.5 shows this example.

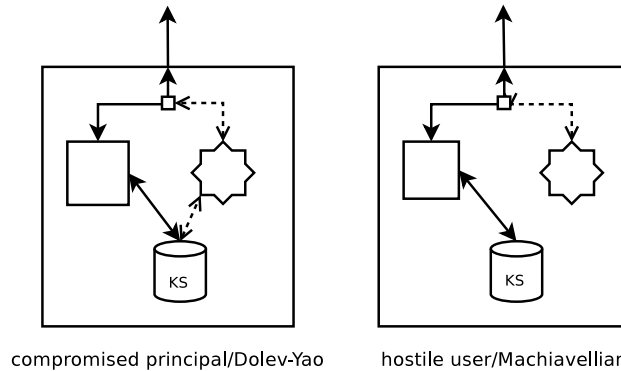
#### 4.4.1.2 Capabilities

One integral aspect of any adversary model, is the capabilities of the adversary which may dictate what the specific adversary may compute, or in other way deduce.

When studying a system to analyse the assumption made with regards to an adversary, the capabilities of the adversary often fall into three different classes of certainty.

**Guaranteed Capabilities** These are concrete and definitive capabilities, the adversary is known to have. For example, in the Dolev-Yao model the adversary’s capabilities are explicitly bound in his actions by the algebraic properties of the protocol in use.





**Figure 4.5:** An example of degrees of compromise in a principal

**Probabilistic Capabilities** Capabilities that have well defined probabilities associated with them are here called probabilistic capabilities. This is often used within computational complexity theory, where the upper and lower bounds may be set for the probability that the adversary has a certain capability, given for example a certain amount of computational resources. In literature on such probabilistic adversaries within cryptography, these are often expressed in terms of *notions* of what an adversary may do (e.g. [7, 48]).

**Possible Capabilities** Capabilities that the adversary may or may not have, are often seen in adversary models based on human adversaries. Such probable capabilities are prevalent in situations where the assumptions with regards to the adversary may not be made with absolute conviction (e.g. [128]).

Capabilities may be many different things. In an anonymity network setting we may assume that an adversary of the “observer” type, will likely have the capability of counting messages and creating a history of observed messages which later may be used for statistical analysis.

#### 4.4.1.3 Resources

These are objects which the adversary may control or has in his possession. These objects may be anything from exploitable persons to access to computational hardware (computational ability), but also more intangible resources such as time are essential.

Resources may also often comprise knowledge the adversary may have, primarily about the victim system.

Depending on the required level of abstraction, the various details of an adversary’s resources may be defined, and they are often strongly connected to the adversary’s capabilities. For example, for the adversary to be capable of monitoring and storing traffic in an anonymity network setting, it will need storage capacity.

#### 4.4.2 Intra-Adversary Channels

An adversary may be working alone, or he may be conspiring with other adversaries of potentially different character. In the case of several cooperating adversaries, they may have access to different channels, and combining their areas of influence may produce an adversary who is several orders of magnitude more dangerous than any adversary operating on his own.

For any number of conspiring adversaries, a channel between each of them are required. Let's call this the intra-adversary channel. This special type of channel has the same definition as the channel between the principals presented in section 4.2, and it is characterised by the operations the adversaries may do to it, as specified earlier in this section.

The intra-adversary channel may, as with the channels between principals, appear in many different shapes and forms.

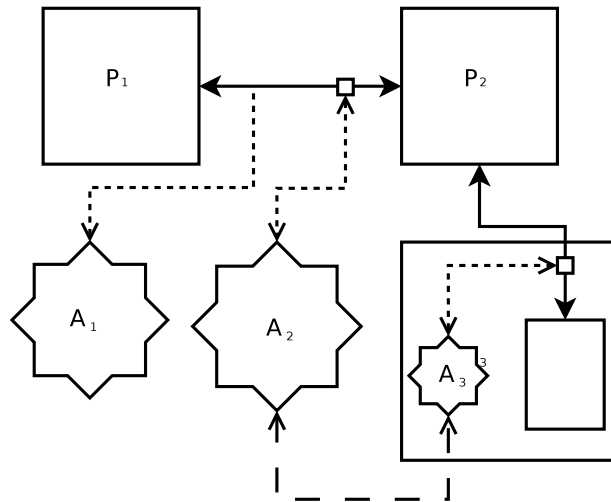


Figure 4.6: Intra Adversary Channels

Figure 4.6 shows how the intra-adversary channel is modelled; as a dashed line. The compromised principal's adversarial element,  $A_3$  has here established a connection to  $A_2$ .

The entire adversarial setting of this system, comprises the adversaries ( $A_1$ ,  $A_2$  and  $A_3$ ) and the intra-adversary channels between them.

## 4.5 Framework Summary and Modelling Pointers

Below follows some minor pointers as to how the framework is thought to be used when modelling a system.

### 4.5.1 Summary of Notation

Figure 4.7 shows a list of the notation used in the models.

### 4.5.2 Modelling Procedure

Although this section is not meant to be normative, the modelling of systems usually consists of three distinct phases:

1. Determine principals and the channels connecting them.
2. Identify the existing adversary model based on assumptions made with regards to the adversary.
3. Identify adversaries not protected against.

Each of these require a certain amount of knowledge of the system's design. Step 1 require the modeller to understand the general data flow of the system. In step 2, the







adversaries	
principals	
channels	
adversary: read	
adversary: write	
adversary: intercept	

Figure 4.7: Summary of Framework Notation.

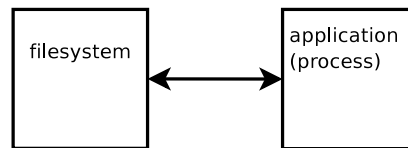


Figure 4.8: Example of Modelling the Interaction Between a Process and the File system

key information is to find what kind of security measures are put in place in the system, and what kind of adversaries they thwart. As for the third step, the key is looking at the unprotected channels identified in step 1 and determining what operations, if any, it is feasible that an adversary may be able to execute against them.

### 4.5.3 Modelling Special Cases

Often, it is not obvious how to model a complex or detailed case using only the primitives presented earlier in this chapter. Below follows some pointers as to how this may be dealt with.

#### 4.5.3.1 Compromised Principals

A compromised principal may be seen as a principal with an internal adversarial element. This element may be anything from a part of the principal's psyche, or a virus or trojan having infected a principal. This makes sense, as even a thoroughly evil participant in a system will still be a principal, i.e., embody e.g. logic or behaviour required to be a principal as it is only principals who may interact.

This was shown on figure 4.5 in section 4.4.1.1.

#### 4.5.3.2 Internal Systems

The modelling style of this framework can operate on arbitrary abstraction levels. To model the operation of a computer program, one might for example model the communication channels between the process of the program and the operating system's different parts. A model does not have to be in absolute accordance with reality, as long as the conceptual model makes sense in a real system with regards to the adversary and his abilities. An example from a symbolic link attack (see section 5.4.1) can be seen in figure 4.8. Here, the adversary has access to the conceptual channel between the application process and the file system of the OS.



## 5 Applying the Framework

A framework of any kind can not be considered successful unless it has been tested on a fairly large set of cases. This chapter will briefly introduce different areas of applied information security, before going on to apply the framework to cases within the areas.

### 5.1 Steganography

Steganography is often described as the art and science of hiding messages in such a way that their presence will pass undetected by any observer not knowing what to look for. In other words, steganography differs from cryptography in the way that secrets are hidden within other messages, rather than being obscured. The steganographic messages are often further secured by using cryptography, but the fact that it is undetectable is the primary attribute. Conversely, the main goal of a steganalyst is to determine whether or not a message contains an embedded, hidden message. Determining the meaning of the embedded message, provided one is found, is not necessary to label the current steganographic scheme as broken.

Steganography is an ancient art which have been practised for a very long time. The result is that there exists a lot of steganographic techniques. In ancient days, they might have used invisible ink, special characteristics in the handwriting for certain letters, and other creative methods. Today, in the digital world, messages are often embedded by harnessing redundant data in for example images to host the message. A rudimentary way of hiding a message is to use the least significant bits of the colour value of individual pixels in a bitmap image. The visible impact on the image will be non-existent, however using this approach may make it trivial for an adversary to discover the hidden message.

The first treatment of steganography in modern scientific literature may have been Simmons who in 1983 formulated what he calls the “prisoners’ problem” [108]. Simmons provides a prison analogy where two inmates are trying to communicate an escape plan. However, all their correspondence must go through the warden, who will take action if any such communication is taking place. To communicate without being thrown into solitary confinement, the prisoners will have to find ways to hide their true communication within some cover communication. This is done through what Simmons calls a “subliminal channel”—a steganographic, limited-bandwidth channel.

In steganography, secret messages are embedded into what is called a *cover object*. The resulting object is a *stego object*, which is then sent over the unsecure channel, as shown in figure 5.1. Often, the algorithm ( $S$  in figure 5.1) which embeds the secret message into the cover object usually takes a key, which is an integral part in making the algorithm adhere to Kerckhoffs’ second principle (see section 2.4). The resulting stego object should be indistinguishable from the cover object by any eavesdroppers. By using the same key

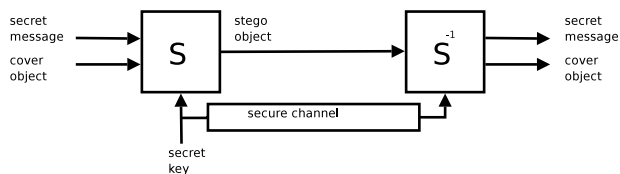


Figure 5.1: The Steganographic Process.

and the inverse steganographic algorithm,  $S^{-1}$ , the secret message is recoverable.

According to [18], steganographic literature mainly identifies two different adversary models: the passive adversary and the active adversary. Usually, the passive adversary is seen when dealing with classic steganography, and the active is usually seen in relation to specific uses of steganography—such as watermarking. The adversary will then already know that an object is a stego object.

### 5.1.1 Classic Steganography

The passive adversary is an adversary with unlimited computational powers, who monitors a communication channel, looking for transmissions or messages between two principals. If a message is detected, the passive adversary some times (even though the adversary is classified as “passive”) has the capability of suppressing the message.

As with cryptography, there have been developed formal frameworks and methods for steganography and steganalysis which draws heavily on Shannon’s work in the field of cryptography (e.g. [4, 16, 131]). In [16], a steganographic technique is deemed “perfectly secure” if the probability distribution of cover objects (messages where hidden messages are embedded) equals the probability distribution of stego objects (the hidden messages). However, due to the non-random nature of messages with embedded hidden content, applying a formal framework to steganography is not as easily done as with cryptography. Anderson and Petitcolas argue in [4] that a perfectly secure steganographic technique (a technique which provides “perfect covertness”) mirroring the perfectly secure one-time pad in cryptography is impossible, which is also found in [131], where they show that for the steganographic technique to be perfectly secure, the cover object and stego object would have to have the same entropy (the uncertainty of the transmitted message) which is true when the cover object equals the stego object.

Let us consider a classic steganographic setting; the prisoners’ problem. Assume that to communicate, two prisoner’s must do so through the warden of the prison. They solve the problem by employing steganography. We see this case from the prisoners’ point of view, thus the warden is our adversary.

**Principals** Two principals communicating.

**Channels** Steganographic techniques will conceptually open up a limited-bandwidth channel between the two principals parallel to the main channel, unbeknown to the warden. See figure 5.2.

**Protected Asset** Using the angle of an adversarial warden, the protected asset in this case is the existence of a channel between the inmates, and thus the information flowing over this channel; an escape plan.

**The Adversarial Setting**

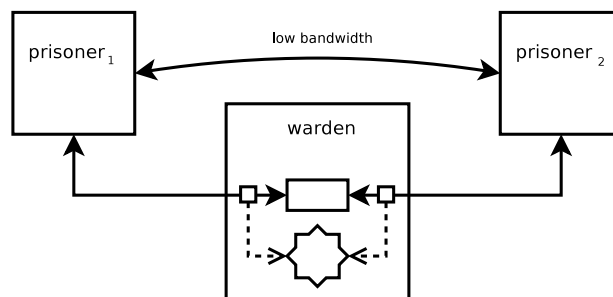


Figure 5.2: Adversary Model of a Classic Steganography Scenario.

**Adversary  $A_1$**  — The warden. Controller of legal communication.

**Channel Operations** *Read* and *intercept* operational access to the channel.

**Capabilities** Can inspect communication, and will recognise illicit communication. The illicit communication can be suppressed from the channel.

**Resources** The warden must have the resources to monitor and read the information channel. Do not need resources to store uninteresting communication.

## 5.2 Digital Rights Management

As mentioned in the previous section (5.1), steganography defines two different main adversary models.

In addition to the passive, an active adversary [88] is also defined, which is mostly used in relation to watermarking and fingerprinting of intellectual property. An adversary will then, in many cases, know beforehand that a hidden message or watermark is embedded into the intellectual property (which may for example be an image) by steganographic techniques. The active adversary is therefore assumed to be able to deliberately alter the message containing the hidden message in an attempt to destroy and/or remove the embedded watermark. It is often assumed that the active adversary may only be able to alter the message in such a way so that the original meaning is not lost. As an example, there would be no point in destroying or severely alter the look of a watermarked image while trying to remove the watermark.

Here, we'll use an example of Digital Rights Management (DRM) [65], which uses steganography to watermark music files with copyright information. DRM often consists of both a digital certification scheme to ensure that only certified users (i.e., those who have purchased the copy) can play the music, but also a digital watermarking scheme which aims to make it possible for a music distributor to prove that the music file indeed has been stolen. Additionally, DRM provides the intellectual property owner ways to control the dissemination of the intellectual property.

Typically, DRM systems are organised as depicted in figure 5.3 [69]. Content providers (usually the intellectual property owner) provides the music to the distributor, who distributes music to its customers, e.g. via a Web interface. The customer is then required to pay the clearinghouse to obtain a digital license with which he is able to play the music file. The clearinghouse then pays the distributor for the distribution and royalties to the content provider. Often, the clearinghouse and distributor are the same entity.

Let us in our case consider the clearinghouse and the distributor to be the same entity.

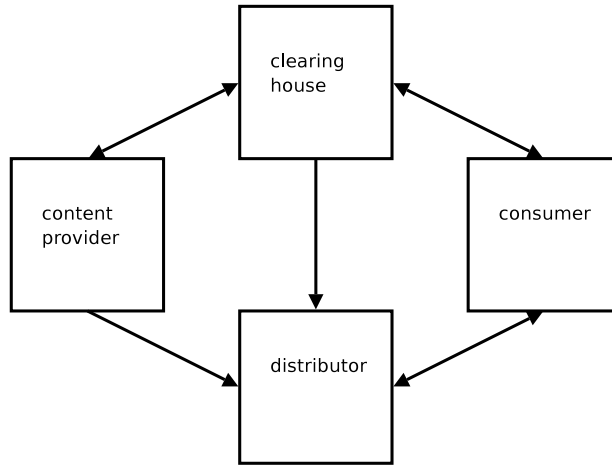


Figure 5.3: Digital Rights Management.

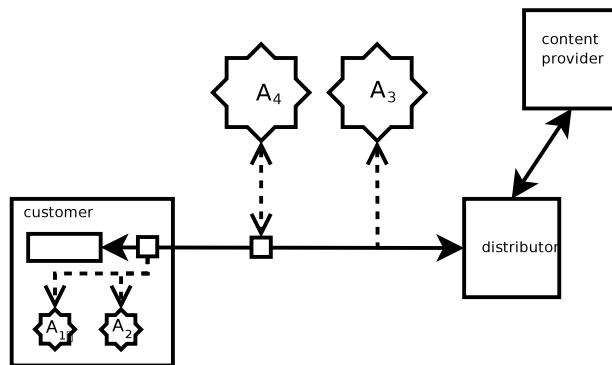


Figure 5.4: Adversary Model of a Digital Rights Management Scenario.



**Principals** Music distribution outlet and the customer purchasing music online (the consumer). Also involved is the intellectual property owner.

### Channels

**Protected Asset** The DRM system aims to prevent theft (i.e., illicit copying) of intellectual property belonging to the music distributor. As such, the value of the intellectual property is the asset. By copying the files, its value will diminish.

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to remove the watermarking.

**Channel Operations** Read (may obtain the intellectual property from the 'legitimate' user).

**Capabilities** Aware of the existence of the watermarking. Has sophisticated capabilities for manipulating the music file in such was that the watermarking will be removed, yet keep the musical information roughly intact.

**Resources** Watermark-removing techniques may be sophisticated and require fairly much computing power, however a modern home-computer would in most cases suffice.

**Adversary  $A_2$**  — An adversary able to remove certification scheme.

**Channel Operations** Read (may obtain the intellectual property from the 'legitimate' user).

**Capabilities** Such as adversary  $A_1$ , this adversary also has sophisticated capabilities which includes reverse engineering the certification scheme, so that it may be removed from the file and subsequently to be able to copy it to others.

**Resources** This adversary do not require much resources, apart from space to store the the files and computing power to remove the certification scheme, once the removal procedure is developed.

**Adversary  $A_3$**  — An adversary capable of monitoring the communication between distributor and customer (see 5.9.2, adversary  $A_1$ ).

**Adversary  $A_4$**  — An adversary capable of masquerading as a distributor (see 5.9.2, adversary  $A_2$ ).

## 5.3 Onion Routing Networks

We will here attempt to apply our framework to the Onion Routing adversarial setting as laid out by Syverson et al [118].

Figure 5.5 shows an example of a anonymity network consisting of tree mixes ( $M$ ), of which one is subverted by an adversary and has become a compromised mix. The network has three users ( $U_1$ ,  $U_2$  and  $U_3$ ), of which one is a hostile user.  $U_2$  and  $U_3$  are on the same local area network, having two proxies to interface with the Internet; one Onion proxy ( $P_2$ ) and one normal proxy ( $P_3$ ). Other adversaries present are one observer ( $A_3$ ) and one disruptor ( $A_4$ ). The adversaries present may use standard communication protocols over the Internet as intra-adversary channels, indicated by the cloud.

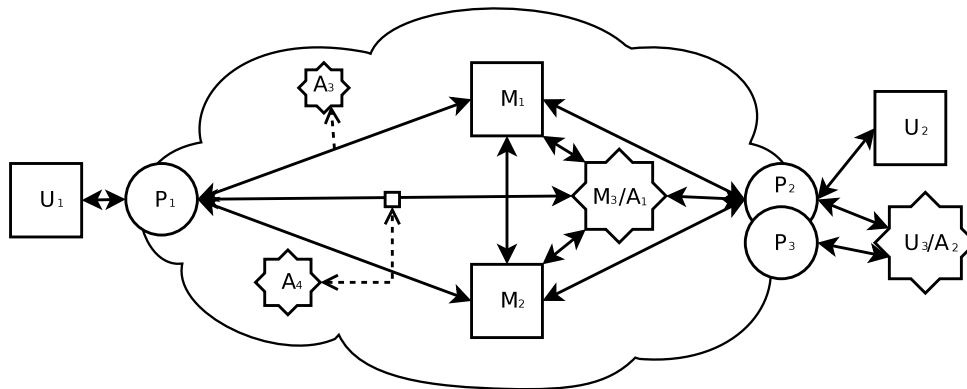


Figure 5.5: Anonymity network adversary model overview.

Note that in the aforementioned figure, the compromised principals have been drawn as adversaries with the principal's channels connected for lack of space and better readability.

**Principals** Users ( $U_n$ ), onion proxies ( $P_n$ ), core onion routers (mixes,  $M_n$ ).

**Channels** Internet-based channels exist between all core onion routers, between the users and the proxies.

**Protected Asset** The protected asset of the anonymity network is usually twofold. Firstly the anonymity of two communicating parties is sought to be protected from prying eyes and traffic analysis methods. Secondly, anonymity networks may also protect the identity of two communicating parties from each other.

In addition many networks, such as the Onion Routing network, provide confidentiality of the transmitted messages.

### The Adversarial Setting

**Adversary  $A_1$**  — The compromised core onion router.

**Channel Operations** *Read, write* and *intercept* operations towards the channels which it is adjacent to.

**Capabilities** Decrypt one layer of the onion by using its private key. Can store package information (where it came from and where it is destined next).

**Resources** Quite large storage resources are required to store most of the traffic passing by the COR.

**Adversary  $A_2$**  — The hostile user.

**Channel Operations** *Read, write*.

**Capabilities** Can send traffic over specific routes in the onion net. May vary the flow of traffic to make it easier to detect patterns.

**Resources** Standard resources of a home computer.

**Adversary  $A_3$**  — The observer.

**Channel Operations** *Read* the channel between either a proxy and a COR, or two CORs.

**Capabilities** Count packages.

**Resources** Requires storage for packages and package counting/monitoring.

**Adversary**  $A_4$  — The disruptor.

**Channel Operations** *Write, intercept* channel(s) between either a proxy and a COR, or two CORs.

**Capabilities** Halt or delay traffic, making it easy to spot a pattern other places in the onion net. Corrupt data travelling over the channel.

**Resources** Standard.

**Intra-Adversary Channels** Cooperating adversaries are considered the only truly effective type of adversary in an anonymity network. For example, a global compromised COR adversary is defined when every mix of a mix-net is compromised. Given such an adversary, it will be able to totally break the anonymity traits of the network. Other configurations of cooperating compromised mixes may also be able to learn very much about who is communicating with whom, and even compromised mixes paired with observers may be able to learn important information, although somewhat dependent on the implementation of the anonymity network. A single compromised COR will only be able to gain information about where messages come from and where they are destined to next.

Given that the Onion Routing network is functioning via a protocol on top of the common Internet protocols (TCP/IP), one may assume that the channels between adversaries are made up of the same network connections, only using a different protocol. It is not detailed in which manner the adversaries are cooperating, but it is assumed that the adversaries may communicate in real time or near real time (with low latency).

## 5.4 Access Control/Local Attacks

### 5.4.1 Symbolic link Attack

A good example of a race condition attack [125, chapter 9], is the old vulnerability in the UNIX `passwd` utility first shown by Bishop and Dilger [9].

The `passwd` utility is shown to be vulnerable, because it creates a temporary file with a predictable file name and doesn't do proper checks with regards to the contents of the file. The adversary may replace the temporary file with a symbolic link to another file, making `passwd` write to the symbolically linked file instead. As `passwd` is run with the effective user ID of the superuser, any file can be written to.

Let us try to analyse the original paper by Bishop and Dilger and see if we are able to find which assumptions they have made with regards to the adversary.

**Principals** The file system and the running application (the process).

**Channels** Conceptually, there exists a channel between the process of the application and the file system, allowing the application read and manipulate files on the system it has access to.

**Protected Asset** The integrity of files on the system. The attack may give the adversary full access to the victim system, compromising whatever is of value on the system or accessible via the system.

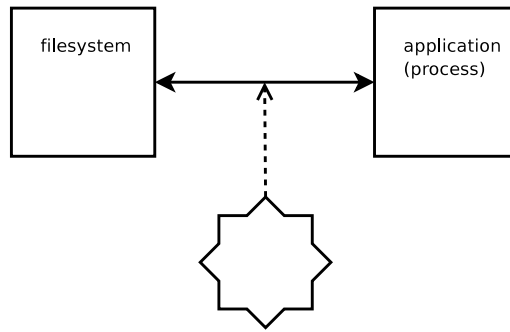


Figure 5.6: Adversary Model of a System Under an Application Vulnerability Attack

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to change files.

**Channel Operations** Conceptually, we can model this adversary as one having *write* operations to the channel between the application and the file system.

**Capabilities** The adversary is capable of executing a timed attack, i.e., having somewhat more than script-kiddie level of sophistication<sup>1</sup>. The adversary must have in-depth knowledge of the Linux or Unix operating system running the vulnerable application. Has the capability of running the exploit over and over until it works, as it may not work on the first try since the window of opportunity is very small.

**Resources** Standard computational resources available. Must have access to the victim computer running the vulnerable application.

This attack may be hard to execute, however utilising techniques such as timed coloured Petri nets [25] to help with the timing of the attack, execution becomes fairly feasible for a local adversary.

## 5.5 Single Sign-On

Single Sign-On (SSO) systems [87] usually provide the user a way to log in once to several services which each usually would require its own log in procedure to access.

To explain SSO, de Clercq [29] uses three fundamental concepts: *authentication infrastructure* which consists of *authentication servers* which are the physical machines performing the authentication, and *authentication authorities*—a logical concept describing entities trusted by users to perform authentication. In an SSO environment, the user would authenticate to an authentication server which is a part of the authentication authority. If the authority finds that the login credentials are correct, the user will be issued a token. The services requiring authentication he wishes to use, will then trust the authentication authority to have correctly authenticated the user, hence allowing access upon display of the token.

An SSO system may appear in many guises. For example, the ticket based authentication system Kerberos [14, 113], is a type of SSO system which enables a user, typically in

<sup>1</sup>A script-kiddie is a term used on inexperienced, non-sophisticated and childish aspiring crackers, see [http://en.wikipedia.org/wiki/Script\\_kiddie](http://en.wikipedia.org/wiki/Script_kiddie) (last visited June 30th, 2005) for more

an enterprise setting, to log onto a workstation, and then transparently being authenticated to any service that the user has access to. The token in this case is the ticket issued by the ticket granting server, which again is part of Kerberos' authentication authority. Other, simpler, systems may rely on cookies in Web browsers as tokens to authenticate users across services offered by Web applications [98].

There are some objections to SSO systems. Often login credentials a user must initially provide is a username-password combination. One of the key objections to SSO systems is that if an adversary manage to gain access to the username and password of a user, he will have access to everything the user has, i.e., he has obtained the "keys to the entire kingdom". Further attacks on SSO systems include an adversary's possibility to forge a token, thereby circumventing the entire authentication process. This immediate access to a potentially wide array of services puts strong demands on the strength of the initial authentication procedure.

### 5.5.1 Kerberos

This discussion about Kerberos is based on [14], which in an intriguing manner details the principles of Kerberos version 4, now long superseded by version 5 [62].

Kerberos was originally designed to provide an authentication service in a distributed environment, where the users were using thin clients and the services such as heavy applications, mail and printing was run on separate servers. The authentication procedure of Kerberos is detailed in figure 5.7.

As previously mentioned, Kerberos revolves around cryptographic tokens called "*tickets*". First, the user requests a ticket-granting ticket from the authentication service. The response, encrypted using the user's password, contains among other things the ticket-granting ticket and a session key to be shared between the ticket-granting service and the user. The user will then request a service ticket from the ticket-granting service by providing the ticket-granting ticket. Upon providing the service ticket to a service, it will respond with an authenticator, which allows the user to use the service for one session.

Steps 1 and 2 shown on figure 5.7 are done once per log on session, 3 and 4 once every type of service required, and finally, step 5 and 6 are done once per service session. The user will only have to sign on once; to provide a password to decrypt the response given in step 2.

The Kerberos authentication scheme seeks to foil an adversary which masquerades as a valid user of a service. One alternative in a distributed system is to allow all the services to know each of the users' passwords. This however, is less than optimal solution and would require a lot of work when the password needs to be changed. Introducing such an elaborate system will introduce new threats and as such new assumptions about adversaries than what is found in simple password systems where all services know the passwords of all users. This is reflected below.

Note that as Kerberos is an authentication scheme, it takes no care as to whether the data interchange between clients and services are secured in any way.

**Principals** The following principals exist in a environment with Kerberos-enforced authentication: The system's clients, the services used by the clients, the ticket-granting service and the Kerberos server.

**Channels** As seen in figure 5.8, there are network channels between clients, services and the Kerberos server.

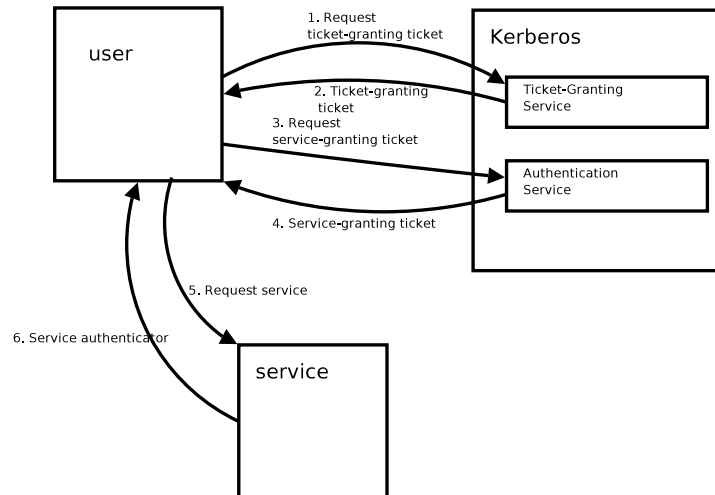


Figure 5.7: The Kerberos Authentication Scheme

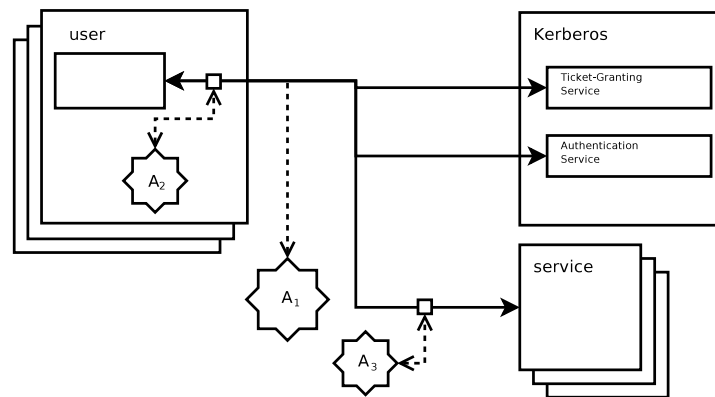


Figure 5.8: The Adversary Model of the Kerberos Authentication Scheme

**Protected Asset** The goal of the Kerberos authentication scheme is as stated that services may only be used by authorised users. As such, Kerberos protects the identity of the users, the identity of the services and it protects the services from unrightful use or abuse.

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to monitor the communication channels.

**Channel Operations** *Read.*

**Capabilities** Has the capability of reading data off of the network as it passes by. This goes for both passwords (pre-Kerberos) and tickets (after Kerberos implementation). The Kerberos protocol seeks to prevent theft of tickets by employing DES encryption. As such, it is assumed that the adversary does not have the capability to break DES, at least for the duration of a typical ticket-granting ticket (note that Kerberos 5 do not put restrictions on encryption algorithm). It is assumed that the adversary has capability to spoof workstation identification, or use a victim's workstation.

It is further assumed that the adversary is capable of replaying retrieved tickets.

**Resources** Receive and store network traffic of interest.

**Adversary  $A_2$**  — An adversary able to masquerade as a client towards a service.

**Channel Operations** *Read, write and intercept.*

**Capabilities** Can replay network traffic, such as tickets and passwords.

**Resources** The user's resources; a terminal or workstation.

**Adversary  $A_3$**  — An adversary able to masquerade as a service towards a client.

**Channel Operations** *Read, write and intercept.*

**Capabilities** Spoof service identity and address. Kerberos guards against this by encrypting the service-granting ticket with the service's key (this is known by the ticket-granting service).

**Resources** Not much resources are needed for this adversary.

### 5.5.2 Mobile Phone-Based Personal SSO System

Kerberos is a prime example of a corporate-level authentication system revolved around single sign-on. On a more personal level several solutions exist, such as the method using cookies for Web service authentication as mentioned earlier. At NISlab, Sverre Moe [78] is working on a single sign-on solution based on a mobile phone with Bluetooth connectivity. It is designed to allow the users to only remember one password. The solution requires a single sign-on proxy which stores the user's passwords for different services. The proxy is operated by autonomous software residing on the mobile phone. Figure 5.9 outlines the design of the system.

This single sign-on solution makes use of the Bluetooth wireless communication protocol. Bluetooth uses a proprietary encryption algorithm referred to as  $E_0$  based on SAFER+ [72]. This algorithm has been under high scrutiny by the cryptographic community and has been found to provide less security than advertised [56, 67]. Recent research has also proven that there are flaws in the protocol used when pairing (interconnecting) two Bluetooth devices. This leads to unconditional compromise of the PIN codes used by most devices when pairing [103] provided the attacker is close enough (i.e., within the range of Bluetooth), and that the PIN is short enough (most phones use 4 digit PINs). Such compromise will at the very least allow an adversary to eavesdrop on the communication.

Bluetooth-enabled devices have also been plagued by implementation flaws leading to such attacks as "Bluebugging" and "Bluesnarfing"<sup>2</sup>, both of which may lead to potential compromise of private data on specific devices. Self-replicating malware (i.e., worms such as Cabir<sup>3</sup>) also exist which use the Bluetooth communication ability of the victim as a way to find new victims to infect. This, however, is not due to deficiencies in the Bluetooth technology per se.

Below we will apply the framework to identify potential adversaries to the solution.

**Principals** The user with his phone and SSO proxy, and the services he uses the SSO solution for.

<sup>2</sup>Both "Bluebugging" and "Bluesnarfing" are described in more detail at <http://www.bluetooth.com/help/security.asp> (last visited June 30th, 2005)

<sup>3</sup><http://www.f-secure.com/v-descs/cabir.shtml> (last visited June 30th, 2005)

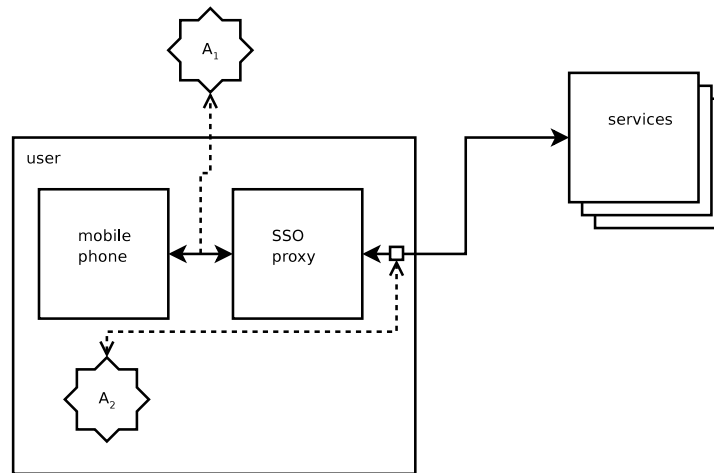


Figure 5.9: A Personal Single Sign-On System for Mobile Phones

**Channels** There is a Bluetooth channel between the mobile phone and the SSO proxy. Further there are HTTP channels between the SSO proxy and the services utilised.

**Protected Asset** Any security measures in this case will guard against the release and compromise of passwords.

#### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to monitor the Bluetooth connection.

**Channel Operations** *Read*.

**Capabilities** Considering the discovery of pairing PIN compromise [103], this adversary is feasible, although the findings in [103] may not directly lead to communication compromise at current time. Capability of executing cryptographic operations, leading to compromise of the PIN used when pairing.

**Resources** Access to somewhat sophisticated radio equipment for intercepting and processing Bluetooth signals. Must be within 10 meters of the victim, or within Bluetooth range.

**Adversary  $A_2$**  — An adversary able to manipulate HTTP data.

**Channel Operations** *Read*, *write* and *intercept* operations on the channel between SSO proxy and services.

**Capabilities** This is a typical trojan adversary, which may manipulate the HTTP data as it enters or leaves the user. The capabilities include understanding the HTTP protocol. This may lead to compromise of the passwords used in the exchange between the SSO proxies and the services.

**Resources** The user's computational resources.

The other possible adversaries in this scenario relate to Web security, presented in section 5.9. As these are not directly related to this type of SSO implementation they are described there.



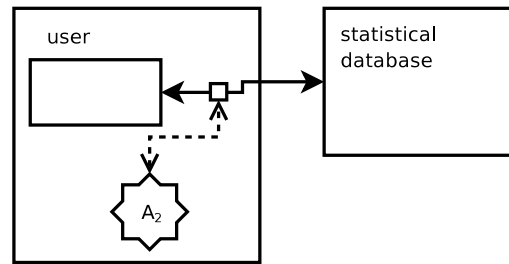


Figure 5.10: Adversary Model of a Database System Under Statistical Attack

## 5.6 Database Security

### 5.6.1 Statistical Attacks

Books on database security (e.g. [17]) deal mostly with access control, which is a major part of database security. In statistical databases (large databases used as empirical base material for research), another type of attack is prevalent; the inference attack.

Consider a case where a database contains highly sensitive data, e.g., medical details of a large set of people. Such databases may be extremely useful in research as empirical data. However, to preserve the privacy of the persons who are listed in the databases, only queries containing data aggregating functions (e.g. SUM, COUNT, AVG, etc.) are allowed.

However, by clever use of aggregating queries, knowledgeable adversaries may still be able to infer sensitive information about only one of the listed persons.

To foil this attempt at breaching privacy of the involved parties, assume that a database administrator has implemented the following countermeasures [49, section 14.4]:

- Aggregate functions must aggregate over minimum three rows in the database table.
- Randomise column entries which will not affect the validity of the empirical data.
- Separate sensitive information and put it in a separate table inaccessible to the user allowed to do statistical analysis.
- Meticulous logging and query analysis.

The assumptions made with regards to the adversary are easily enumerated using the framework:

**Principals** The database system and its users.

**Channels** Typical network channels exists between the principals.

**Protected Asset** The confidentiality of the sensitive material contained in the medical database, which again may have immense value for many different buyers.

#### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to access the database with statistical attack capabilities.

**Channel Operations** The adversary is a likely adversarial user of the database, and as such has conceptually *read* and *write* access to the channel between the database and the user.

**Capabilities** Understands the SQL specifications and protocols used to communicate with the database. Knowledgeable within the following areas:

**direct attack** An aggregate is calculated over a small sample by using a predicate (in SQL this is usually the WHERE clause in the statement) known to be common to only a few entries

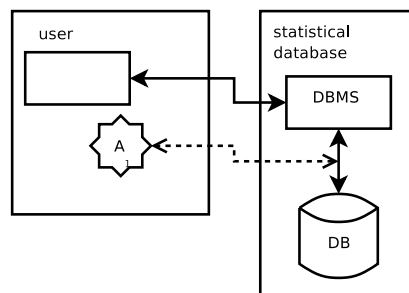
**indirect attack** Combining several direct attacks to infer information about an entry not possible with only direct attacks

**tracker attack [32]** An attack based on finding a predicate which can be used as a “key” in aggregate statements to allow the retrieval of information about one row in the database table, usually by way of several database queries.

The adversary already has access to the database (as a statistician).

**Resources** Standard resources available. Knowledgeable in SQL and relative simple statistics.

There is also an alternative way of seeing this adversary model, namely as a deficiency in the DBMS as the adversarial statistician circumvents countermeasures enforced by the DBMS. This is shown in figure 5.11.



**Figure 5.11:** Alternate View of the Adversary Model of a Database System Under Statistical Attack

## 5.7 Wireless Networks

This section gives an opportunity to see how the framework helps in highlighting the differences between two different adversary models for the same scenario, namely wireless 802.11 networking.

### 5.7.1 The Wired Equivalent Privacy (WEP) Security Protocol

Wireless networks are inherently exposed to eavesdropping, and WEP was introduced as an optional part of the 802.11 specification [54, section 8.2] to prevent what the specification calls “casual eavesdropping”.

WEP is implemented using a stream cipher as shown in figure 5.12. The stream cipher (Rivest’s RC4) is seeded by an initialisation vector and a secret key from which the output is XOR-ed with the plaintext and its integrity checksum (CRC-32).

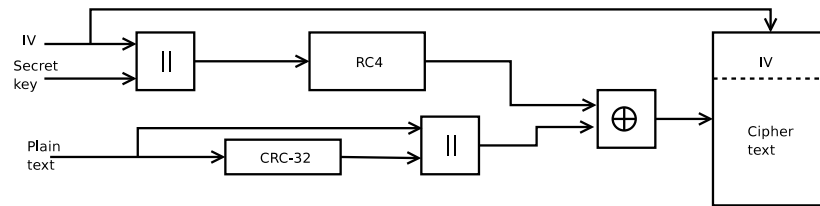


Figure 5.12: The Wired Equivalent Privacy Security Protocol (based on [54])

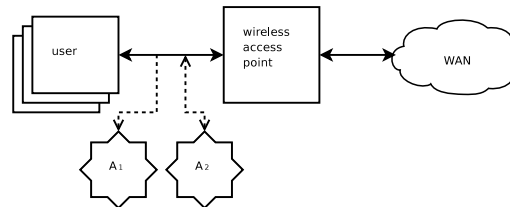


Figure 5.13: Adversary Model of a Wireless Network.

Although WEP ultimately was shown to be totally unsecure (see section 5.7.2 and [11, 115]), let's have a look at the assumptions made with regards to the adversary, which prompted the introduction of WEP.

Figure 5.13 shows the adversary model of a wireless network as the designers of WEP probably saw it.

**Principals** The principals in a wireless networking scenario are the users of the wireless network, and the wireless access point.

**Channels** The obvious channels here are the radio based channels over which the users communicate with the access point.

**Protected Asset** WEP was introduced to safeguard the general confidentiality and integrity of the data travelling over the network. Also, part of its job is to keep anyone from connecting to the network, that is provide authentication and access control.

#### Adversarial Setting

**Adversary  $A_1$**  — An adversary able to eavesdrop on the radio frequency channel.

**Channel Operations** The adversary is able to *read* the radio frequency channels between principals.

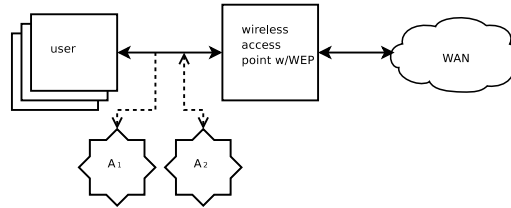
**Capabilities** Receive data in the radio frequency spectrum (reception antenna).

**Resources** Capable radio receiver to monitor signals.

**Adversary  $A_2$**  — An adversary able to influence the information transmitted on the radio frequency channel.

**Channel Operations** The adversary is able to *write* to the radio frequency channels between principals.

**Capabilities** Send data in the radio frequency spectrum (transmission antenna).



**Figure 5.14:** Adversary Model of a Wireless Network with WEP

**Resources** Capable radio transmitter to manipulate signals.

Adversary  $A_1$  is foiled by implementing fairly strong encryption (RC4 with 64 or 128 bits long key, but of which 24 bits are the open IV). The threat from adversary  $A_2$  is mitigated by using a checksum, or digest (CRC-32).

### 5.7.2 WEP—Post-Break

Unfortunately, WEP is a prime example of how good and sound cryptographic primitives used in a wrong way will compromise security. RC4 is abused and CRC-32 is wrong tool for the job in an environment where the digest must be cryptographically secure. As a result, WEP is still vulnerable to these same adversaries shown in the previous section.

WEP is supposed to ensure similar degree of security as is given by typical copper-wire cabling for wireless networks.

Taking the findings about WEP by Borisov et al [11] into consideration, the adversary model of the WEP is in fact quite different. They found that

- An adversary may circumvent the integrity check, as CRC-32 is a linear algorithm (modifying data so that the CRC checksum is equal is not difficult enough). Thus, the adversary may both inject messages and tamper with messages. The lack of proper integrity checking may in fact also lead to breach in confidentiality.
- The same seed (key) is submitted to RC4 too often, making simple cryptanalytic attacks possible as the same key stream is applied several times. Thus, an adversary may compromise the confidentiality of messages.

Thus, the adversaries identified in the previous section, which the WEP protocol was thought to protect against are back, albeit with slightly different capabilities and resources. As the principals, channels and the protected asset are the same as in the previous section, it will suffice to describe the adversarial setting here.

#### Adversarial Setting

**Adversary  $A_1$**  — An adversary able to eavesdrop on the radio frequency channel.

**Channel Operations** The adversary is able to *read* the radio frequency channels between principals.

**Capabilities** Receive data in the radio frequency spectrum (reception antenna). Also, the adversary is capable of perform data gathering to obtain messages using the same key stream (identified by equal initial vector).

The adversary is then capable of performing an XOR operation on the intercepted messages,  $C_1 \oplus C_2 = M_1 \oplus M_2$ , where  $C$  is obtained ciphertext and  $M$  is cleartext equivalent. The adversary is also capable of solving  $M_1 \oplus M_2$  with regards to both  $M$  (e.g., by applying methods from [28]).

**Resources** Capable radio receiver to monitor signals.

**Adversary  $A_2$**  — An adversary able to influence the information transmitted on the radio frequency channel.

**Channel Operations** The adversary is able to *read* and *write* to the radio frequency channels between principals.

**Capabilities** Send and receive data in the radio frequency spectrum (reception/transmission antenna). The adversary has the capability to obtain messages, then modify them by exploiting the linearity of CRC-32. It is then possible for an adversary to craft a message  $C'$  that decrypts to  $M'$  where  $M' = M \oplus \Delta$ , and  $\Delta$  may be arbitrarily chosen (to match whatever change the adversary wishes to introduce). This message may then be reintroduced onto the network.

Further, the adversary has the capability to introduce totally arbitrary message onto the system. He can do this because most wireless access points are set up to authenticate users by sending out an unencrypted challenge. Upon reception of the encrypted challenge, the user is authenticated. This can be used to learn the keystream used in the challenge response, as  $C \oplus P = K$ , because  $C = P \oplus K$ . As there usually are no restrictions on keystream reuse and the integrity check is unkeyed, the adversary may easily create a message  $v||C_{adv} = (M_{adv}||c(M_{adv}) \oplus K)$  (where  $v$  is the initial vector of  $C$ ,  $||$  indicates concatenation and  $c(M_{adv})$  indicates calculating the CRC-32 checksum of  $M_{adv}$ ), which he may introduce onto the network.

**Resources** Capable radio transmitter to manipulate signals.

Software exists which can find a WEP key in seconds or minutes, such as Kismet<sup>4</sup>.

## 5.8 Authentication Systems

### 5.8.1 Biometrics

Best practises for testing biometric devices are based on zero-effort attacks [70], that is attacks where the adversary does not make any efforts into fooling the biometric authentication system. Often, this is the only adversary model many manufacturers assume and research has shown that fake fingers which are accepted by fingerprint systems are easily created [73, 127].

Uludag and Jain identifies eight different possible ways to attack biometric authentication systems [122]. Among these we find attacks against the other channels of the system, than just the sensor.

The adversary model of many fingerprint recognition devices, can be described as follows.

<sup>4</sup><http://www.kismetwireless.net/> (last visited June 30th, 2005)

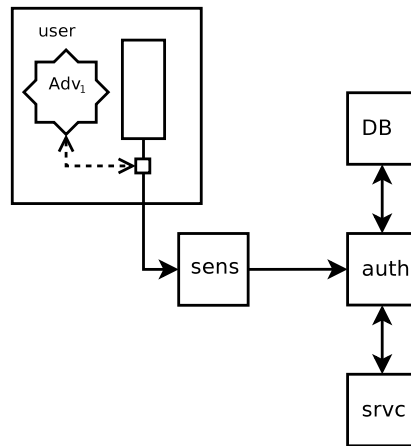


Figure 5.15: Zero-Effort Biometric Authentication Adversary

**Principals** Biometric sensor and the authentication system. Storage of biometric templates. The service which requires authentication.

**Channels** We find channels between the principals as shown in figure 5.15. The channel between the sensor and the authentication service is usually a USB networking channel. The other channels are usually

**Protected Asset** The fingerprint system acts as an authentication method protecting access to something of value.

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to try to slip in as a different user.

**Channel Operations** The adversary has *read*, *write* and *intercept* access to the channel between the valid principal and the sensor device.

**Capabilities** It is assumed that the adversary will only attempt zero-effort attacks. I.e., he is not capable of manufacturing fake fingerprints to present to the sensor device.

**Resources** No resources are required except the finger to present to the device.

Based on the figure of current adversary models (figure 5.15), we can see that there are several unprotected channels in the system. Some of these channels may have realistic threats in forms of adversaries. Also, we can assume that adversary  $A_1$  from above has some extended capabilities. As principals, channels and the protected asset remains the same, the following only focus on the new adversarial setting of the new adversary model. The new adversary model is visualised in figure 5.16.

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to create fake fingerprints.

**Channel Operations** The adversary has *read*, *write* and *intercept* access to the channel between the valid principal and the sensor device.

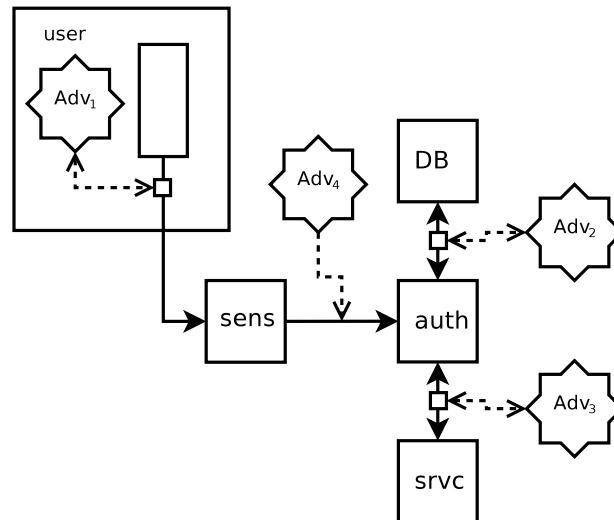


Figure 5.16: Active Biometric Authentication Adversary

**Capabilities** The knowledge and capability to create artificial fingerprints. Good knowledge and experience in fingerprint-lifting methods.

**Resources** Effects needed to create artificial fingerprints, such as gelatin, silicone, fine powder, etc. More on this subject can be seen in [127].

**Adversary  $A_2$**  — An adversary able to manipulate the template retrieval

**Channel Operations** *Read, write and intercept.*

**Capabilities** Capable of communicating with both template database and authentication system, and being able to spoof or manipulate content travelling over the channel.

**Resources** Must know how the minutiae extraction of the fingerprint system works, to be able to create a spoofed fingerprint template which will be accepted by the authentication system.

**Adversary  $A_3$**  — An adversary able to manipulate authentication messages to the service

**Channel Operations** *Read, write and intercept.*

**Capabilities** The adversary is capable of delivering forged approval of identify messages to the service, tricking it to believe a certain individual is authenticated. The adversary is capable of communicating with the service (i.e., he knows the protocol).

**Resources** Knowledge of the protocol is as mentioned a resource this adversary will possess.

**Adversary  $A_4$**  — An adversary able to manipulate the channel over which digitised fingerprints travel

**Channel Operations** *Write and intercept.*

**Capabilities** Capable of injecting changed or completely new digitised fingerprint for delivery to the authentication system, thus circumventing the fingerprint sensor altogether. We do not here assume that the adversary

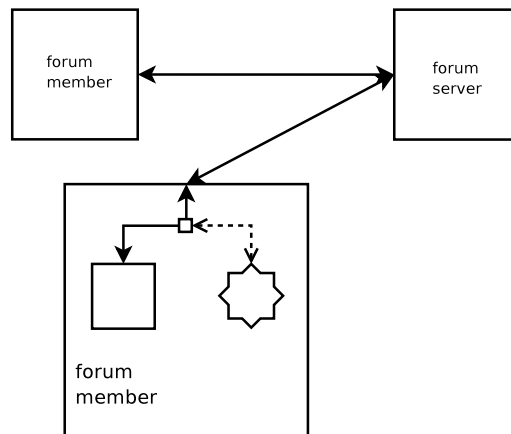


Figure 5.17: Adversary Model of a password-protected Web forum.

is capable of injecting or manipulating the USB wire between sensor and authentication system. To manipulate the digitised fingerprint, the adversary must know the format.

**Resources** As many fingerprint sensors communicate with the authentication system via USB, a simple pre-programmed USB dongle may be sufficient to deliver a digitised fingerprint. As some sensors require that data travelling between sensor and authentication system be encrypted, the adversary may need to know certain encryption keys.

## 5.9 Web Application Security

As the Internet and the World Wide Web proliferate, more and more services are being made available as remote Web applications. Web applications are increasingly becoming part of everyone's daily life, hence it is a very important area.

In this section, we will apply our framework to different kind of Web applications scenarios, which has different requirements with regards to security, and thus the assumptions they make with regards to potential adversaries.

Application security [53, 39] is also an important consideration, although out of the scope of the following example cases.

### 5.9.1 Simple Login Sites

Many sites on the Web has user-specific adaptations, and many also allow the users of the site to contribute to it. One example of such a Web application is what is commonly known as Web forums, or discussion boards.

On the vast majority of such sites, the users' personae are protected by passwords only.

**Principals** In this case, we have to kinds of principals; the forum debater, and the forum server itself. There are several debaters.

**Channels** There will be a communication channel based on HTTP and TCP/IP between the forum server and each of the debaters currently online.

**Protected Asset** The asset is the persona of the forum debaters. A compromise could make adversaries post using the persona of the victim.



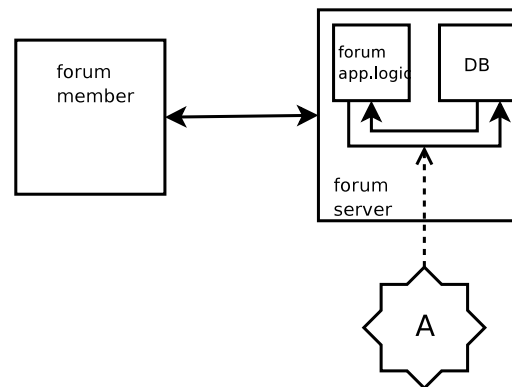


Figure 5.18: Forum Server under SQL Injection Attack.

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to masquerade as a debater.

**Channel Operations** This adversary may be seen as an adversarial element of a legitimate forum debater, and as such he has *read*, *write* and *intercept* access to the channel between the legitimate element of the entity and the forum server (see figure 5.17).

**Capabilities** The adversary must know when to substitute its own username with that of the victim, in order to post messages as the victim. The adversarial part of the legitimate forum debater does not need a more thorough understanding of the protocol in use, as the legitimate whole of the debater will inhabit those.

**Resources** In terms of resources, the adversary may not need to have more than average household computing power.

Obviously, there are a number of other types of adversaries an online forum service may be exposed to, however the security measures exhibited by the server in this case, tells us that the only adversary assumed to be of any risk, is the one which may try to use the persona of a debater.

For example, the adversaries sending malicious data over the communication channel with an intent to exploit or probe for weaknesses are extremely commonplace. The forum software phpBB<sup>5</sup>, have had a fair share of such vulnerabilities<sup>6</sup>. Figure 5.18 shows an adversary performing a typical SQL injection attack [53] against the forum server. Conceptually, we may see this as an unintended *write* access to the channel between the SQL server and the forum server.

### 5.9.2 eCommerce

This case is based around a classical electronic commerce setting where the merchant's electronic commerce solution is based on WWW connectivity, i.e. HTTP<sup>7</sup> over TCP/IP. To ensure confidentiality and integrity of sensitive data the customer may transfer, such as credit card information, the communication channel is secured by using TLS<sup>8</sup>, which

<sup>5</sup><http://www.phpbb.com> (visited 30th June 2005)

<sup>6</sup><http://secunia.com/product/463/> (visited 30th June 2005)

<sup>7</sup>Hyper Text Transfer Protocol [42]

<sup>8</sup>Transport Layer Security [33]

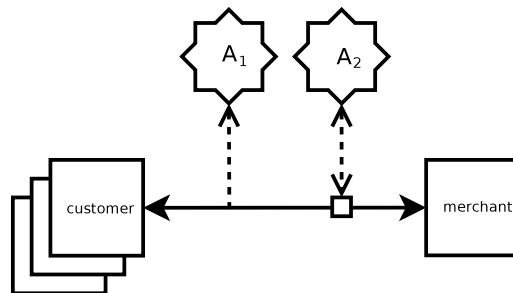


Figure 5.19: Adversary Model of Electronic Commerce

encrypts data flowing over the channel in real time. The authenticity of the merchant is ensured by a certificate.

By introducing a TLS-secured channel, the merchant is able to foil a set of different adversaries. The difference in security between a TLS-secured server and a server without is twofold:

- The information travelling over the channel between the customer and the merchant server are no longer transmitted in clear text. Its integrity is also assured.
- The server authenticates itself towards the customer by the PKI solution provided by TLS. It is based on certificates and a set of certificate authorities known to all Web browsers.

In addition, the eCommerce solution uses usernames and password to authenticate the customers.

**Principals** There are at least two principals in this scenario; the ecommerce site, and its customer. In most cases there are several customers.

**Channels** There will be a TLS-tunnelled communication channel over HTTP and TCP/IP between the server and each of the customers.

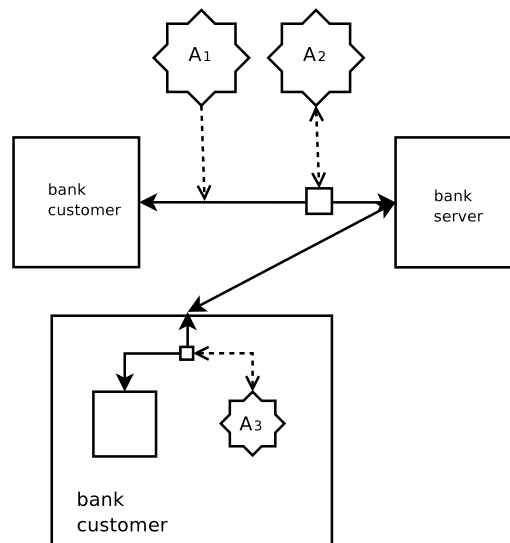
**Protected Asset** The asset here is the confidentiality and integrity of sensitive data regarding purchase and payment travelling between the customer(s) and the merchant.

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to monitor the communication channel.

**Channel Operations** The fact that the merchant takes usage of TLS into consideration, indicates that an adversary is able to *read* the information as it is transmitted over the channel between the customer and the merchant server.

**Capabilities** This adversary understands the HTTP protocol and the data format used by the merchant server for information exchange. Thus, the adversary able to discern login, purchase and payment data from the information flow. The adversary is not assumed to be able to break strong symmetric and asymmetric encryption as used by TLS for securing the information channel.



**Figure 5.20:** Adversary Model of Online Banking with Mutual PKI-based Authentication

**Resources** The amount of resources assumed to be available to the adversary is that of an average customer.

**Adversary  $A_2$**  — An adversary able to masquerade as a merchant server towards the customer.

**Channel Operations** Being able to masquerade as the merchant server, the adversary is able to conceptually *read*, *write* and *intercept* information travelling over the customer-merchant channel. While the actual implementation of such an attack can be done in many ways (DNS poisoning, local Windows HOSTS file modifications, router control), the conceptual implementation in this framework requires the aforementioned operations.

**Capabilities** The adversary is knowledgeable of the protocol in use between the merchant server and the customer, in such a way that he is able to fool the customer into believing he is communicating with the actual merchant server.

**Resources** In terms of resources, the adversary may not need to have more than average household computing power.

### 5.9.3 Online Banking

As a Web application, online banking has very much in common with our previous example, the online merchant.

In this case, we'll look at an online bank solution which uses both server side and client side certificates, to ensure mutual authentication, as the Norwegian online bank Skandiabanken. As with the merchant in the previous example, our Web bank application is communicating by using HTTP over TCP/IP, secured by TLS.

**Principals** There are at least two principals in this scenario; the online bank site, and its customer or customers.

**Channels** TLS-tunnelled communication channel over HTTP and TCP/IP

**Protected Asset** The asset here is the confidentiality and integrity of sensitive data regarding the customer's or customers' bank and personal details. Also underlying is the value of the customer's ability to transfer money without allowing unauthorised persons to do so.

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary capable of monitoring the information travelling over the communication channel. See section 5.9.2, adversary  $A_1$ .

**Adversary  $A_2$**  — An adversary capable of masquerading as the bank's server. See 5.9.2, adversary  $A_2$ .

**Adversary  $A_3$**  — An adversary capable of masquerading as a bank's customer.

**Channel Operations** *Read*, *write* and *intercept* on the channel between the principal and the bank.

**Capabilities** Can figure out simpler authentication mechanisms and talks/understands the protocol.

**Resources** Standard computational resources.

### 5.9.4 eVoting

With the drop in election participation in many of the Western democracies, electronic voting—Internet-based voting in particular—is often seen by governments as a means to increase participation back to old heights.

Naturally, the security demands of such a system must be extremely high. A security breach may very well jeopardise the democracy of a nation. As a result, information security professionals strongly advise against any such adoption of technology [81], at least when it comes to elections of national importance.

The reason for their contempt towards electronic voting is often more rooted in the complexity of the systems which will inevitably lead to bugs and security flaws, than in the underlying design and protocols.

In the case of electronic voting, the scenario is usually the old way of voting being replaced by using a computer with a touch screen. These machines are usually called DREs, which is an abbreviation for the rather awkward "Direct Recording Electronic" machine. As the name indicates, the ballots are counted electronically upon vote. The hope is that this will reduce invalid ballots and counting errors. Internet voting on the other hand, allows the voter to place his vote from the confines of his own home. In this scenario, it is more difficult to control the conditions under which the vote is given. However, close inspection of the source code of a DRM has also showed an unfortunate amount of flaws [63], making the amount of perceived control one can have over DRM-based elections lessen.

As on many other areas, Chaum pioneered work on election protocols [19, 21] for use over networks. Chaum identified three important requirements (in [21]) for such a protocol:

- A voter's privacy can not be violated.

- Voters can ensure that their ballot has been counted.
- Voters cannot disrupt the election process.

Since then, more required properties of election protocols has been identified. Chen et al propose a secure and anonymous voting system for the Web [22] which has the following properties:

- **Fairness.** The outcome of the election is not known until the votes are counted.
- **Eligibility.** Only eligible voters may attend the election.
- **Uniqueness.** One vote per voter.
- **Uncoercibility.** There is no way the voter can show or in other ways prove that he voted a particular way. This is to prevent bribery and selling of votes.
- **Anonymity.** At the tally, there is no link between the voter and the vote.
- **Accuracy.** The protocol ensures that valid votes are counted properly.
- **Efficiency.** All computations can be performed within reasonable time.
- **Robustness.** Adversarial voters can not sabotage the election.
- **Mobility.** No restrictions are being laid on the location of the voter. A Web environment is envisioned.
- **Practicability.** No extra skills are required to vote. Depending on the implementation, this may be debatable though.

Note that Chaum's requirements of 1988 are mirrored in the properties *anonymity*, *accuracy* and *robustness*.

Other similar protocols for large scale distributed systems, e.g. the Internet, also inhabit most of these properties [36, 45, 58].

Figure 5.21 shows the structure of the Chen et al scheme. Its principals besides the voter are the authentication centre which manages voter authentication and eligibility checking, and also provides the voter with a pseudonym signature to apply to the vote. Then there's various public proxy servers used as anonymising proxies which forwards the voter's ballot to both a supervisor centre and a tally centre. The supervisor centre monitors the tallying.

This system has much in common with the previous Web-based systems discussed in this section. Here, however, rather than having communication directly between the client (voter) and the server (tallying centre), an intermediate proxy server has been introduced to separate votes from network identity (IP and non-scrubbed protocols).

The full overview of Chen et al's Internet voting protocol applied in our framework is seen in figure 5.22.

**Principals** Voters, Anonymising proxies, Certificate/Authentication Authority, Tally Centre, Supervising Centre

**Channels** Channels exists between the principals as shown in figure 5.22.

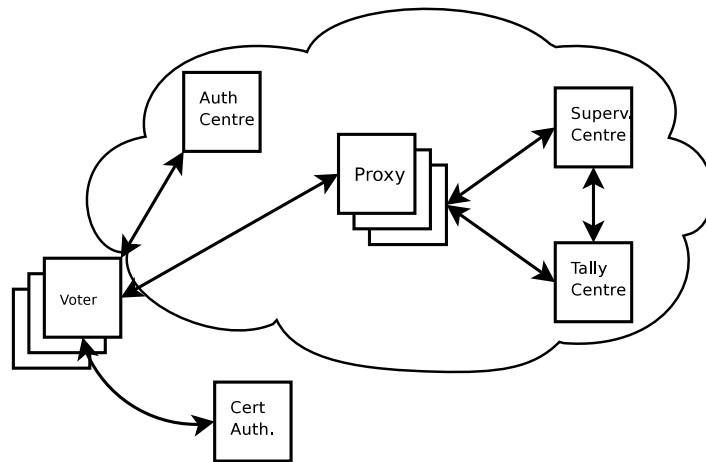


Figure 5.21: The Internet Voting System of Chen et al [22]

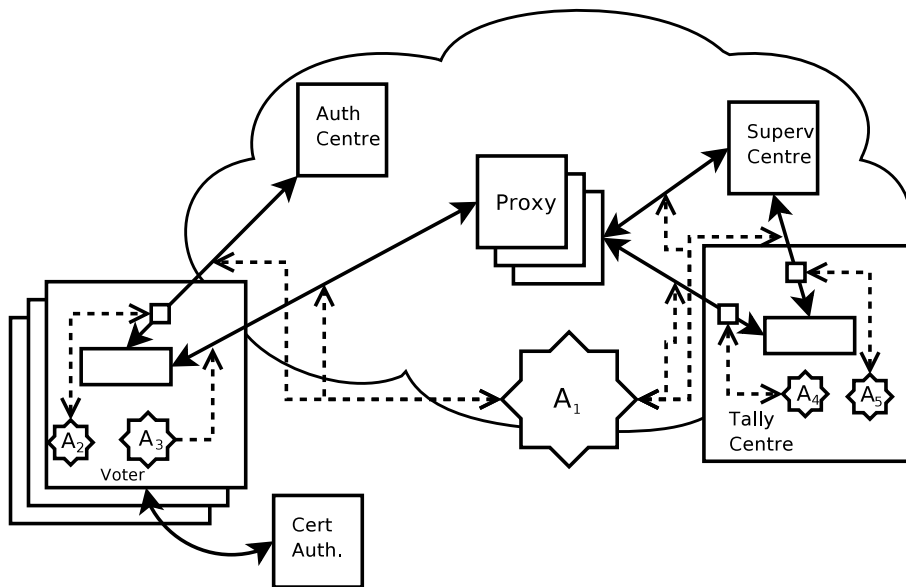


Figure 5.22: Adversary Model of Chen et al's Voting Scheme Over the Internet

**Protected Asset** The confidentiality, integrity and anonymity of the vote.

### The Adversarial Setting

**Adversary  $A_1$**  — An adversary able to read Internet-based communication channels.

**Channel Operations** This adversary has the ability to *read* any of the Internet-based communication channels between the principals. As seen by figure 5.22, these channels are many and might not be accessible to any one real-life adversary. However, it makes sense to group this adversary together in this case. The intra-adversary communication capabilities when talking about Internet entities are probably so high, that they might appear as one adversarial entity.

**Capabilities** Understands the protocol being used. Can not break RSA.

**Resources** Standard resources available. Physical access to the medium (router along the path).

**Adversary  $A_2$**  — An adversary able to masquerade as a voter.

**Channel Operations** *read, writer, intercept*

**Capabilities** Impersonate voter. Knows the protocol. This type of adversary is protected against by using public key certificate (X.509), and as such it is assumed that he cannot break RSA.

**Resources** Standard resources.

**Adversary  $A_3$**  — Voter trying to disrupt the election by voting multiple times.

**Channel Operations** *Read, write and intercept* on the channel between the voter and the proxy ( $V_2$  and P in figure 5.22).

**Capabilities** Cast vote by using protocol knowledge. The system protects against this adversary by RSA, thus he is not assumed to be able to break RSA.

**Resources** Standard resources available.

**Adversary  $A_4$**  — An adversarial tally centre, leaking intermediate voting results.

**Channel Operations** *Read* the channel from the proxies.

**Capabilities** Able to release the ballot counts, i.e. it has a possibly covert channel out of the tally centre.

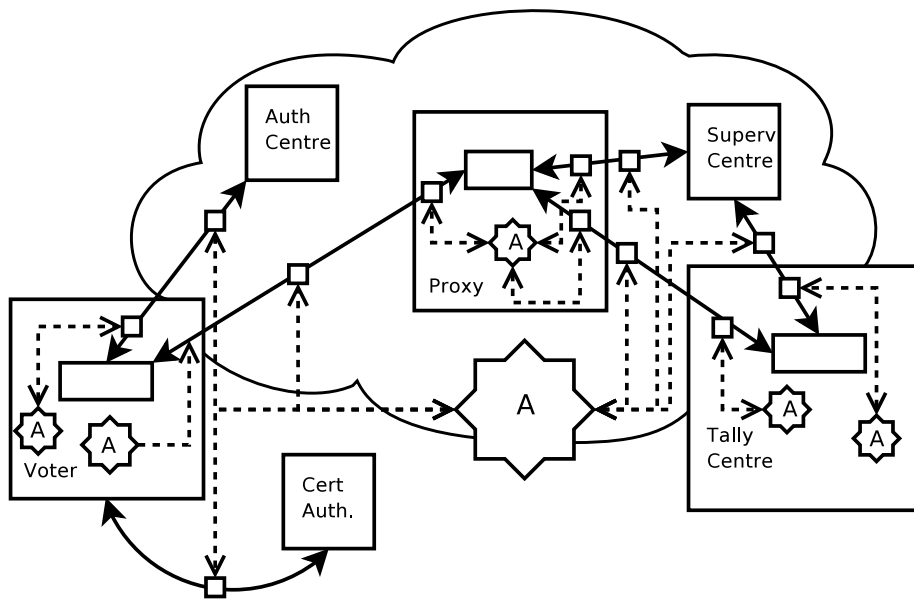
**Resources** As this adversary is a part of the tally centre, it does not need substantial resources on its own. nor will it use much of the tally centre's.

**Adversary  $A_5$**  — An adversarial tally centre, miscounting ballots.

**Channel Operations** *Read, write and intercept* operations towards the channel between the proxies and the tally centre. Will intercept and/or create new data on the channel, to skew the count.

**Capabilities** Understanding of the protocol in use. Is protected against by heavy encryption, and the supervisor and tally centres can only count the votes together.

**Resources** As the adversary's goal is to only drop votes, it does not need much resources. However, if the goal is to drop specific votes, resources are required to store lists over which voters' votes should not be counted.



**Figure 5.23:** Extended Adversary Model of Chen et al's Voting Scheme Over the Internet

As the voting in this case is going to take place over an inherently unsecure medium, namely the Internet, it is easy to envision, and not unrealistic that there exists, adversaries with extended access to channels. The framework allows us to easily see unprotected channels in Chen et al's proposed protocol. In figure 5.23, we have extended the adversary model given by Chen et al to incorporate other plausible adversaries.

Interesting to note, there are quite a few adversaries Chen et al's scheme do not protect against. It does for example not assume that an adversary may influence the distribution of certificates to the voters. A typical distribution channel for certificates would be the postal services, to which several accomplishable attacks exists, e.g., typical mailbox attacks<sup>9</sup>. Also, it does not protect against disruptive attacks such as any forms of attack which will effectively intercept traffic between voters and proxies, and between the proxies and the tally and supervisor centres. Also, the responsibility of protecting the anonymity of the voter's network communications lies fully on the anonymising proxies. Dishonest proxy maintainers or flawed server configurations may compromise the anonymity of the voter, and dishonest tally and supervisor centres may in concert with dishonest proxy maintainers fully recover which way a certain voter voted.

Obviously, many of these attacks are not feasible to prevent in a cryptographic protocol, but are important aspects to consider before implementing a large-scale election based on this scheme.

<sup>9</sup>An example of a "mailbox attack" is when the adversary lies in wait by the victim's mail box and steals the mail as it is delivered by postal workers. Alternatively an adversary may ally himself with a dishonest insider at the post office or elsewhere in the delivery chain.



## 6 Discussion

The main goal of our framework was to facilitate a quick and efficient way to realise and visualise the assumptions made with regards to the adversary or adversaries a system may be exposed to. Often, the assumptions made with regards to an adversary are characterised by the lack of a coherent and collected set of information—something the presented framework aims to remedy.

Considering the cases shown in chapter 5, the framework has proven itself to help gathering these pieces of information into a coherent whole. During the development of the cases, the framework was used extensively in the identification of adversary assumptions. The framework helped keeping a keen eye on the lookout for important information with regards to the security countermeasures embedded in the system. The information could then later be plotted into figures and the text-oriented part of the framework.

The framework—via figures and text—helps highlight differences in assumptions in cases of similar nature, such as the case where the assumptions made with regards to the adversary before and after the break of WEP (section 5.7) are highlighted.

A further strength of the framework is that the data flow diagram-like approach to modelling enable analysts to clearly see channels which are exposed to adversaries not yet protected against. This is easily seen after the initial modelling of assumptions as information channels where no adversary is assumed. Such a case may be an indication of a ignored type of adversary, or simply a very inaccessible channel, not assumed to be reachable by an adversary. The latter cases are rare.

As to the case of the ignored adversaries, there may be several reasons for these to have been left out. Among sound reasons we may find low risk, unfeasible exploitation, prohibitive costs and multitudes more. Other times, however, adversaries may have been ignored due to ignorance or oversight.

It is in this case the framework makes it easy for a customer of a system to gauge the expediency of a system's security countermeasures in a specific environment. Some times, adversaries seen as unlikely by the developers of the system, might not be as unlikely seen from the customer's point of view. Also, adversaries overlooked by the developer may be perfectly obvious to the customer. The framework helps identify these adversaries.

Given the framework's ability to pinpoint assumptions with regards to adversaries, it may also be helpful in the early phases of a system's life cycle. That is, in the design and implementation phases it may help to give an initial overview of the dangers that threat the system.

The experience gathered from using the framework on the cases presented in the previous chapter is good. To model the system's principals and information flows, a certain level of knowledge is required. However, depending on the level of abstraction chosen, specific in-depth knowledge of the system in question is usually not required. For a customer considering implementations from several contractors or suppliers this is beneficial. The same applies to the modelling of the adversary models; a certain modicum of knowledge about the built-in security countermeasures is required, however a mostly superficial knowledge is adequate. This will be enough to be able to visualise and reason about the system.

## 7 Future Work

The framework has in this thesis been applied to a set of different cases. To be able to gauge the efficiency and expediency of the framework, it needs to be applied to more cases. Especially in areas which are not reducible to a client-server networking scenario, such as the symbolic link attack on `passwd` as shown in section 5.4.1.

Some ideas (of which some more explored than others) for additional cases include;

**Automatic payment of toll roads** How is security in different payment systems for roads? Is it possible to model the physical security of the device they put in cars?

**The electronic hospital** With the proliferation of networked and digitalised communication within hospitals, it would be interesting to take a closer look at the assumption designers of such systems have made.

**Business to Business (B2B) services** The communication and procurement strategies of businesses is another interesting area to explore. Lately, B2B activities on the World Wide Web has become more and more commonplace [26, 46], and it would be interesting to see if these have different adversary models as opposed to the other WWW-based services covered here (section 5.9).

**Outsourcing—Application Service Providers** With specially made software and its hardware demands reaching new heights during the past decades, using an Application Service Provider (ASP) has become more and more commonplace. The ASP provides applications with a cost which medium and small businesses would consider prohibitive if bought for a small fee. The applications and the hardware which they depend upon is then made available remotely for the businesses subscribing to the service.

An ASP may be compared to a commercial airline company. As the price of purchasing, owning and staffing a private jet is prohibitive to most, we pay a comparatively minor fee to be able to use the airline's jet. In essence, the airline is here operating in a way very much like an ASP.

Lately ASPs has moved away from proprietary private networks and is increasingly providing their services over the Internet and the World Wide Web [119].

There are a number of security related issues with using ASPs. First and foremost, there is the obvious issue of transferring sensitive data across open networks. Many solves this problem by using Virtual Private Networks (VPNs). However, the issue of confidentiality and integrity of the data when exposed to open channels is not

all. There is also the issue of privacy [13]. That is, the data's confidentiality and integrity with regards to the ASP itself. In some cases, the customer trusts the ASP enough to make this a non-issue, however many smaller businesses may be forced to use an ASP they do not trust due to necessity.

**Web browsers** A very interesting theme is security in Web browsers. Built into most of today's most-used Web browsers, such as Mozilla derivatives<sup>1</sup>, Opera<sup>2</sup> and even Microsoft Internet Explorer are enormous amounts of security countermeasures to protect its users from the vast amount of phishing scams [110], pharming [41], IDN domain spoofs [57] and network monitoring.

Another interesting aspect is that even though Web browsers' security has been under scrutiny for a long time, especially their Java Script-implementation [30], weaknesses are still being discovered.

**Remote Contract Signing** A lot of research have been done on fair contract signing over the Internet [5, 50, 71, 80, 92]. It would be interesting to see what kind of assumptions they have made with regards to the adversary, and also test the framework on a case where both of the contract signing principals may have adversarial elements.

**Malware** Worms is a form of malware which has flourished with the permeation of the Internet and the Web. Recently, we've seen worms such as the Santy worm spreading via Google [66]. However, this is a slow method of propagation.

The Witty worm is one of the most frightening worms yet released on the Internet [64, 126]. It only infected about 12,000 computers, but only because it exploited, and thus specifically targeted, computers running a certain version of BlackIce Defender<sup>3</sup>. It used similar extremely efficient spreading techniques, such as those shown by Staniford et al [111].

What sets the Witty worm apart from other worms and malware is the professionalism of its design. It contains no significant bugs which could hinder it spreading, and it showed a real malicious intent. Within 45 minutes of its release, it has infected all of the roughly 12,000 vulnerable hosts connected to the Internet. Witty was a worm written by a very experienced programmer with an intent to do malice, rather than a kid out for some fun.

It would be very interesting to apply the framework to such a case where the adversary is an extremely rapidly spreading threat.

As the framework also has its uses within system engineering disciplines, it would be very interesting to have a group of system engineers and software engineers use the framework in the initial stages of a system's life cycle. This way we could judge how well the framework is at identifying adversaries and thus threats in an efficient manner.

---

<sup>1</sup><http://www.mozilla.org> (last visited June 30th, 2005)

<sup>2</sup><http://www.opera.com/> (last visited June 30th, 2005)

<sup>3</sup><http://blackice.iss.net/> (last visited June 30th, 2005)

## 8 Conclusions

We have presented a new framework for adversary modelling. The framework has been successfully used on several heterogeneous cases, and has contributed to gaining and overview of and increased insight into the adversarial setting of the systems treated in the cases. As such, it seems that this modelling style reminiscent of data flow diagrams works well.

It is also shown that the framework is suitable for easy identification of adversaries not protected against in the current implementation of the system, such as seen in section 5.9.4. This may be invaluable for customers as they with knowledge of the system's security features model the assumed adversarial setting. By evaluating the remaining information channels possibly available to an adversary, they may determine whether or not the system incorporates adequate security measures.

It is also hoped that the presented framework, alongside forms of threat modelling, can help contribute to the design phases of a system's life cycle by providing a novel way to stimulate the creativity of the designers using a quick graphical modelling method which focuses on the general information flow of the system, as opposed to the much-used detailed data flow diagrams.



## Bibliography

- [1] Martín Abadi. Taming the adversary. In Mihir Bellare, editor, *Proceedings of Advances in Cryptology – CRYPTO 2000: 20th Annual International Cryptology Conference*, volume 1880 of *Lecture Notes in Computer Science*, pages 353–358, August 2000. 2.5
- [2] Martín Abadi and Andrew D. Gordon. A calculus for cryptographic protocols and spi calculus. In *Proceedings of the Fourth ACM Conference on Computer and Communications Security*, April 1998. 3.3
- [3] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *Journal of Cryptography*, 15(2):103–127, January 2002. 3.3
- [4] Ross J. Anderson and Fabien A.P. Petitcolas. On the limits of steganography. *IEEE Journal on Selected Areas in Communications*, 16(4):474–481, May 1998. 5.1.1
- [5] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18(4):593–610, April 2000. 7
- [6] David E. Bell and Leonard J. LaPadula. Secure computer system: Unified exposition and multics interpretation. Technical Report MTR-2997, The MITRE Corporation, May 1976. 4.2
- [7] Mihir Bellare, Anand Desai, Eron Jorjani, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *Proceedings of 38th Annual Symposium of Foundations of Computer Science*, pages 394–403, October 1997. 2.4, 4.4.1.2
- [8] Oliver Berthold, Hannes Federrath, and Stefan Köpsell. Web MIXes: A system for anonymous and unobservable Internet access. In Hannes Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *Lecture Notes in Computer Science*, pages 115–129. Springer-Verlag, July 2000. 2.2
- [9] Matt Bishop and Michael Dilger. Checking for race conditions in file access. *Computing Systems*, 9(2):131–152, 1996. 5.4.1
- [10] Dan Boneh. The Decision Diffie-Hellman problem. In J.P. Buhler, editor, *Proceedings of the Third Algorithmic Number Theory Symposium*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer-Verlag, June 1998. 2.4.4.1
- [11] Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting mobile communications: The insecurity of 802.11. In *MobiCom '01: Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pages 180–189, Rome, Italy, July 2001. ACM Press. 2.4.4, 5.7.1, 5.7.2

- [12] Philippe Boucher, Adam Shostack, and Ian Goldberg. *Freedom System 2.0 Architecture*. Zero-Knowledge Systems, Inc., 18 December 2000. 2.2
- [13] Claus Boyens and Oliver Günther. Trust is not enough: Privacy and security in asp and web service environments. In Y. Manolopoulos and P. Nárvat, editors, *Proceedings of 6th East European Conference on Advances in Databases and Information Systems*, volume 2435 of *Lecture Notes in Computer Science*, pages 8–22, September 2002. 7
- [14] Bill Bryant. Designing an authentication system: a dialogue in four scenes (draft), February 1988. 5.5, 5.5.1
- [15] Michael Burrows, Martín Abadi, and Roger Needham. A logic of authentication. *ACM Transaction on Computer Systems*, 8(1):18–36, 1990. 3.3
- [16] Christian Cachin. An information-theoretic model for steganography. In *Information Hiding*, volume 1525 of *Lecture Notes in Computer Science*, pages 306–318. Springer-Verlag, 1998. 5.1.1
- [17] Silvana Castano, Maria G. Fugini, Giancarlo Martella, and Pierangela Samarati. *Database Security*. ACM Press Books. Addison-Wesley, 1994. 5.6.1
- [18] Rajarathnam Chandramouli, Mehdi Kharrazi, and Nasir Memon. Image steganography and steganalysis: Concepts and practice. In *IWDW 2003*, volume 2939 of *Lecture Notes in Computer Science*, pages 35–49. Springer-Verlag, 2004. 5.1
- [19] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981. 2.2, 5.9.4
- [20] David Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1):65–75, 1988. 2.2
- [21] David Chaum. Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. In C.G. Günther, editor, *Proceedings of EUROCRYPT'88*, volume 330 of *Lecture Notes in Computer Science*, pages 177–182, 1988. 5.9.4
- [22] Yu-Yi Chen, Jinn-Ke Jan, and Chin-Ling Chen. The design of a secure anonymous internet voting system. *Computers & Security*, 23(4):330–337, June 2004. 5.9.4, 5.21
- [23] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer-Verlag, 1998. 2.4.4.1
- [24] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES – the Advanced Encryption Standard*. Springer-Verlag, 2002. 2.4
- [25] Ole Martin Dahl and Stephen D. Wolthusen. Modeling and execution of complex attack scenarios using interval timed colored Petri nets. In *Proceedings of the 21st Annual Computer Security Applications Conference*, Tucson, Arizona, USA, December 2005. Submitted for publication. 5.4.1



- 
- [26] Qizhi Dai and Robert J. Kauffman. Business models for internet-based e-procurement systems and b2b electronic markets: An exploratory assessment. In *Proceedings of the 34th Hawaii International Conference on System Sciences*, 2001. 7
- [27] George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pages 2–15, May 2003. 2.2
- [28] E. Dawson and L. Nielsen. Automated cryptanalysis of XOR plaintext strings. *Cryptologia*, (2):165–181, April 1996. 5.7.2
- [29] Jan de Clercq. Single sign-on architectures. In *Proceedings of InfraSec 2002*, volume 2437 of *Lecture Notes in Computer Science*, pages 40–58, 2002. 5.5
- [30] Flavio de Paioli, Andre L. dos Santos, and Richard A. Kemmerer. Web browsers and security. In G. Vigna, editor, *Mobile Agents and Security*, volume 1419 of *Lecture Notes in Computer Science*, pages 235–256. Springer-Verlag, 1998. 7
- [31] Richard A. DeMillo, Nancy A. Lynch, and Michael J. Merritt. Cryptographic protocols. In *Proceedings of the fourteenth annual ACM Symposium on Theory of Computing*, pages 383–400, 1982. 3.3
- [32] Dorothy E. Denning, Peter J. Denning, and Mayer D. Schwartz. The tracker: A threat to statistical database security. *ACM Transactions on Database Systems*, 4(1):76–96, March 1979. 5.6.1
- [33] T. Dierks and C. Allen. The TLS Protocol Version 1.0. RFC 2246 (Proposed Standard), January 1999. Updated by RFC 3546. 8
- [34] Whitfield Diffie and Martin E. Hellmann. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, November 1976. 2.4
- [35] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004. 2.2
- [36] Gianluca Dini. A secure and available electronic voting service for a large scale distributed system. *Future Generation Computer Systems*, 19(1):69–85, 2003. 5.9.4
- [37] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. *SIAM Journal on Computing*, 45(4):727–784, 2000. A preliminary version appeared in STOC’91. 2.4
- [38] Danny Dolev and Andrew C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, March 1983. 2.5, 4.4.1.1
- [39] André L.M. dos Santos, Giovanni Vigna, and Richard A. Kemmerer. Security testing of the online banking service of a large international bank. In *Proceedings of the First Workshop on Security and Privacy in E-Commerce*, November 2000. 5.9

- [40] Taher Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IT-31(4):469–471, July 1985. 2.4
- [41] Entrust. Phishing is yesterday’s news: Get ready for pharming, April 2005. 7
- [42] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard), June 1999. Updated by RFC 2817. 7
- [43] Rune Fredriksen, Monica Kristiansen, Bjørn Axel Gran, Ketil Stølen, Tom Arthur Opperud, and Theo Dimitrakos. The CORAS framework for a model-based risk management process. In S. Anderson, S. Bologna, and M. Felici, editors, *Proceedings of the 21st International Conference on Computer Safety, Reliability and Security*, volume 2434 of *Lecture Notes in Computer Science*, pages 94–105. Springer-Verlag, September 2002. 3.1
- [44] Michael J. Freedman and Robert Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, November 2002. 2.2
- [45] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques: Advances in Cryptology*, pages 244–251, 1992. 5.9.4
- [46] Jai Ganesh, T.R. Madanmohan, P.D. Jose, and Sudhi Seshadri. Adaptive strategies of firms in high-velocity environments: The case of B2B electronic marketplaces. *Journal of Global Information Management*, 12(1):41–59, 2004. 7
- [47] David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. Hiding router information. In R. Anderson, editor, *Proceedings of the First International Workshop on Information Hiding*, volume 1174 of *Lecture Notes in Computer Science*, pages 137–150. Springer-Verlag, May 1996. 2.2
- [48] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. 2.4, 4.4.1.2
- [49] Dieter Gollmann. *Computer Security*. Worldwide Series in Computer Science. Wiley, 1999. 5.6.1
- [50] Andrew Goodchild, Charles Herring, and Zuran Milosevic. Business contracts for B2B. In *Proceedings of the CAISE’00 Workshop on Infrastructure for Dynamic Business-to-Business Service Outsourcing*, volume 30 of *CEUR Workshop Proceedings*, 12 July 2001. 7
- [51] Ceki Gülcü and Gene Tsudik. Mixing E-mail with Babel. In *Proceedings of the 1996 Network and Distributed Security Symposium*, pages 2–16, February 1996. 2.2
- [52] Martin E. Hellmann. An extension of the Shannon theory approach to cryptography. *IEEE Transactions on Information Theory*, IT-23(3):289–293, May 1977. 2.4

- 
- [53] Sverre H. Huseby. *Innocent Code: A Security Wake-Up Call for Web Programmers*. Wiley, 2004. 5.9, 5.9.1
- [54] IEEE. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification — ANSI/IEEE Std. 802.11, 1999. 5.7.1, 5.12
- [55] ISO/IEC. Information technology: Code of practice for information security management. ISO/IEC 17799, 1 December 2000. 4.3
- [56] Markus Jakobsson and Susanne Wetzal. Security weaknesses in bluetooth. In D. Naccache, editor, *Proceedings of the Cryptographers' Track at RSA Conference*, volume 2020 of *Lecture Notes in Computer Science*, pages 176–191, April 2001. 5.5.2
- [57] Eric Johanson. The state of homograph attacks. Advisory, <http://www.shmoo.com/idn/homograph.txt> (last visited June 30th, 2005), 11 February 2005. 7
- [58] Jared Karro and Jie Wang. Towards a practical, secure, and very large scale on-line election. In *Proceedings of the 15th Annual Computer Security Applications Conference*, pages 161–169, 1999. 5.9.4
- [59] Michelle Keeney, Eileen Kowalski, Dawn Cappelli, Andrew Moore, Timothy Shimeall, and Stephanie Rogers. Insider threat study: Computer system sabotage in critical infrastructure sectors, May 2005. 2.3
- [60] Richard A. Kemmerer. Analyzing encryption protocols using formal verification techniques. *IEEE Journal on Selected Areas in Communication*, 7(4):448–457, May 1989. 3.3
- [61] Auguste Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, IX:5–38, January 1883. <http://www.petitcolas.net/fabien/kerckhoffs/> (last visited June 30th, 2005). 2.4
- [62] J. Kohl and C. Neuman. The Kerberos Network Authentication Service (V5). RFC 1510 (Proposed Standard), September 1993. 5.5.1
- [63] Tadayoshi Kohno, Adam Stubblefield, Aviel D. Rubin, and Dan S. Wallach. Analysis of an electronic voting system. In *Proceedings of the 2004 IEEE Symposium on Security and Privacy*, pages 27–40, May 2004. 5.9.4
- [64] Abhishek Kumar, Vern Paxson, and Nicholas Weaver. Exploiting underlying structure for detailed reconstruction of an internet scale event. Technical report, Georgia Institute of Technology, May 2005. 7
- [65] Sai Ho Kwok. Digital rights management for the online music business. *ACM SIGecom Exchanges*, 3(3):17–24, August 2002. 5.2
- [66] Elias Levy and Iván Arce. Worm propagation and generic attacks. *IEEE Security & Privacy Magazine*, 3(2):63–65, March 2005. 7

- [67] Ophir Levy and Avishai Wool. A uniform framework for cryptanalysis of the Bluetooth  $E_0$  cipher. Cryptology ePrint Archive, Report 2005/107, 11 April 2005. 5.5.2
- [68] P Lincoln, J. Mitchell, M. Mitchell, and A. Scedrov. A probabilistic poly-time framework for protocol analysis. In *Proceedings of the 5th ACM conference on Computer and Communications Security*, pages 112–121, 1998. 3.3
- [69] Qiong Liu, Reihaneh Safavi-Naini, and Nicholas P Sheppard. Digital rights management for content distribution. In *Proceedings of the Australasian Information Security Workshop Conference on ACSW Frontiers*, volume 21, pages 49–58, 2003. 5.2
- [70] Anthony J. Mansfield and James L. Wayman. Best practices in testing and reporting performance of biometric devices. Technical Report CMSC 14/02, National Physical Laboratory, August 2002. 5.8.1
- [71] Hiroshi Maruyama, Taiga Nakamura, and Tony Hsieh. Optimistic fair contract signing for Web services. In *Proceedings of the ACM workshop on XML security*, pages 79–85, New York, NY, USA, 2003. ACM Press. 7
- [72] James L. Massey, Gurgun H. Khachatrian, and Melsik K. Kuregian. Nomination of SAFER+ as candidate algorithm for AES, 12 June 1998. 5.5.2
- [73] T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino. Impact of artificial gummy fingers on fingerprint systems. In *Proceedings of SPIE, Optical Security and Counterfeit Deterrence Techniques IV*, volume 4677, pages 275–289, 2002. 5.8.1
- [74] John P. McDermott. Attack net penetration testing. In *Proceedings of the 2000 Workshop on New Security Paradigms*, pages 15–21, September 2000. 3.2
- [75] Catherine Meadows. A system for the specification and analysis of key management protocols. In *Proceedings of the 1991 IEEE Symposium on Research in Security and Privacy*, pages 182–195, May 1991. 2.5, 3.3
- [76] Catherine Meadows. Open issues in formal methods for cryptographic protocol analysis. In *Proceedings of DARPA Information and Survivability Conference and Exposition*, volume 1, pages 237–250, January 2000. 3.3
- [77] Adel Melek and Marc MacKinnon. *2005 Global Security Survey*. Deloitte Touche Tohmatsu, 22 June 2005. 2.3
- [78] Sverre Moe. Mobile single sign-on. Master’s thesis, Norwegian Information Security Laboratory at Gjøvik University College, 1 July 2005. 5.5.2
- [79] Ulf Möller, Lance Cottrell, Peter Palfrader, and Len Sassaman. Mixmaster protocol version 2. Internet-Draft, July 2003. 2.2
- [80] David Molnar. Signing electronic contracts. *Crossroads*, 7(1):6–ff., 2000. 7
- [81] David P Moynihan. Building secure elections: E-voting, security, and systems theory. *Public Administration Review*, 64(5):515–528, October 2004. 5.9.4

- [82] Steven J. Murdoch and George Danezis. Low-cost traffic analysis of Tor. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 2005. 2.2
- [83] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks, July 1995. Preliminary version appeared in Proceedings of the 22nd ACM Symposium on Theory of Computing (1990). 2.4.4
- [84] National Institute of Standards and Technology. *FIPS PUB 46: Data Encryption Standard (DES)*. National Institute for Standards and Technology, 1988. 2.4
- [85] National Institute of Standards and Technology. *FIPS PUB 197: Advanced Encryption Standard (AES)*. National Institute for Standards and Technology, 26 November 2001. 2.4
- [86] Roger M. Needham and Michael D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978. 2.5
- [87] Andreas Pashalidis and Chris J. Mitchell. A taxonomy of single sign-on systems. In R. Safavi-Naini and J. Seberry, editors, *ACISP 2003*, volume 2727 of *Lecture Notes in Computer Science*, pages 249–264. Springer-Verlag, 2003. 5.5
- [88] Fabien A.P Petitcolas, Ross J. Anderson, and Markus G. Kuhn. Attacks on copyright marking systems. In David Aucsmith, editor, *Proceedings of the Second Workshop on Information Hiding*, volume 1525 of *Lecture Notes in Computer Science*, pages 218–238, April 1998. 5.2
- [89] Birgit Pfitzmann, Matthias Schunter, and Michael Waidner. Cryptographic security of reactive systems (extended abstract). *Electronic Notes in Theoretical Computer Science*, 32, 2000. 3.3
- [90] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology - CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer-Verlag, 1992. 2.4.4.1
- [91] Marisa R. Randazzo, Michelle Keeney, Eileen Kowalski, Dawn Cappelli, and Andrew Moore. Insider threat study: Illicit cyber activity in the banking and finance sector, August 2004. 2.3
- [92] Indrajit Ray and Indrakshi Ray. Fair exchange in e-commerce. *ACM SIGecom Exchange*, 3(2):9–17, May 2002. 7
- [93] Michael Reiter and Aviel Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, November 1998. 2.2
- [94] Marc Rennhard. Practical anonymity for the masses with mix-networks. Technical Report TIK-Nr. 157, Swiss Federal Institute of Technology, February 2003. 2.2
- [95] Marc Rennhard and Bernhard Plattner. Introducing MorphMix: Peer-to-peer based anonymous internet usage with collusion detection. In *Proceedings of the Workshop on Privacy in the Electronic Society*, November 2002. 2.2

- [96] Marc Rennhard, Sandro Rafaeli, Laurent Mathy, Bernhard Plattner, and David Hutchinson. An architecture for an anonymity network. In *Proceedings of the IEEE 10th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 165–170, June 2001. 2.2
- [97] Ronald L. Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, February 1978. 2.4
- [98] Vipin Samar. Single sign-on using cookies for web applications. In *Proceedings of IEEE 8th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 158–163, June 1999. 5.5
- [99] Bruce Schneier. Attack Trees. *Dr. Dobb's Journal*, December 1999. 3.2
- [100] Bruce Schneier. *Secrets and Lies: Digital Security in a Networked World*. Wiley Publishing Inc., 2000. 2.4
- [101] Gregg Schudel and Bradley Wood. Modeling the behaviour of the cyber-terrorist. Submitted to National Information Systems Security Conference, October 2000. 2.1
- [102] Gregg Schudel and Bradley Wood. Adversary work factor as a metric for information assurance. In *Proceedings of the 2000 Workshop on New Security Paradigms*, pages 23–30. ACM Press, 2001. 2.4
- [103] Yaniv Shaked and Avishai Wool. Cracking the Bluetooth PIN. In *Proceedings of the Third Annual International Conference on Mobile Systems, Applications and Services: MobiSys*, June 2005. 5.5.2, 5.5.2
- [104] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27, 1948. 2.4, 2.4
- [105] Claude E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, 1949. 2.4, 2.4, 5
- [106] Eric Shaw, Keven G. Ruby, and Jerrold M. Post. The insider threat to information systems. *Security Awareness Bulletin*, 2, September 1998. 2.3
- [107] R. Shirey. Internet Security Glossary. RFC 2828 (Informational), May 2000. 1
- [108] Gustavus J. Simmons. The prisoners' problem and the subliminal channel. In *Proceedings of CRYPTO'83*, pages 51–67, August 1984. 5.1
- [109] Sean W. Smith. Turing is from Mars, Shannon is from Venus: Computer science and computer engineering. *IEEE Security & Privacy Magazine*, 3(2):66–69, March 2005. 6
- [110] Sophos. Phishing and the threat to corporate networks. <http://www.sophos.com/whitepapers/sophos-phishing-wpuk.pdf> (last visited June 30th, 2005), December 2004. 7

- 
- [111] Stuart Staniford, Vern Paxson, and Nicholas Weaver. How to Own the Internet in your spare time. In *Proceedings of the 11th USENIX Security Symposium*, 2002. 7
- [112] Jan Steffan and Markus Schumacher. Collaborative attack modeling. In *Proceedings of the 2002 ACM Symposium on Applied Computing*, pages 253–259, 2002. 3.2
- [113] Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An authentication service for open network systems. In *Proceedings of USENIX Winter*, 1988. 5.5
- [114] Jacques Stern. Cryptography and the methodology of provable security. In *Proceedings of 15th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, volume 2643 of *Lecture Notes in Computer Science*, pages 1–5, May 2003. 2.4.4.1
- [115] Adam Stubblefield, John Ioannidis, and Aviel D. Rubin. Using the Fluhrer, Mantin, and Shamir attack to break WEP. Technical Report TD-4ZCPZZ, AT&T Labs, 21 August 2001. 5.7.1
- [116] Frank Swiderski and Window Snyder. *Threat Modeling*. Microsoft Professional. Microsoft Press, 2004. 3.1
- [117] Paul Syverson, Catherine Meadows, and Iliano Cervesato. Dolev-Yao is no better than Machiavelli. In *Proceedings of First Workshop on Issues in the Theory of Security*, 2000. 4.4.1.1
- [118] Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an analysis of onion routing security. In Hannes Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *Lecture Notes in Computer Science*, pages 96–114, July 2001. 2.2, 4.4.1.1, 5.3
- [119] Lixin Tao. Shifting paradigms with the application service provider model. *Computer*, 34(10):32–39, October 2001. 7
- [120] The Common Criteria Project. Common criteria for information technology security evaluation part 1: Introduction and general model v2.1, August 1999. 3.1
- [121] A. Turing. On computable numbers, with an application to the entscheidungsproblem. In *Proceedings of London Mathematical Society*, volume 42, 1937. 6
- [122] Umut Uludag and Anil K. Jain. Attacks on biometric systems: A case study in fingerprints. In *Proceedings of Security, Steganography, and Watermarking of Multimedia Contents VI*, volume 5306, pages 622–633, 2004. 5.8.1
- [123] United States Department of Defense. Trusted computer system evaluation criteria. DOD 5200.28-STD, 26 December 1985. 3.1
- [124] Gilbert S. Vernam. Cipher printing telegraph systems for secret wire and radio telegraphic communications. *Journal of the American Institute of Electrical Engineers*, 1926. 2.4

- [125] John Viega and Gary McGraw. *Building Secure Software: How to Avoid Security Problems the Right Way*. Addison-Wesley Professional Computing Series. Addison-Wesley, May 2002. 5.4.1
- [126] Nicholas Weaver and Dan Ellis. Reflections on Witty: Analyzing the attacker. *login.*, pages 34–37, June 2004. 7
- [127] Anders Wiehe, Torkjel Søndrol, Ole Kasper Olsen, and Fredrik Skarderud. Attacking fingerprint sensors. [http://www.olekasper.no/articles/attacking\\_fingerprint\\_sensors.pdf](http://www.olekasper.no/articles/attacking_fingerprint_sensors.pdf) (last visited June 30th, 2005), 15 December 2004. 5.8.1, 5.8.1
- [128] Bradley Wood. An insider threat model for adversary simulation, 2000. 2.1, 2.3, 4.4.1.2
- [129] Andrew C. Yao. Theory and applications of trapdoor functions. In *Proceedings of 23rd Annual Symposium on Foundations of Computer Science*, 1982. 2.4
- [130] Edward Yourdon. *Modern Structured Analysis*. Prentice-Hall International, 1989. 4.2
- [131] Jan Zöllner, Hannes Federrath, Herbert Klimant, Andreas Pfitzmann, Rudi Piontraschke, Andreas Westfeldt, Guntram Wicke, and Gritta Wolf. Modeling the security of steganographic systems. In David Aucsmith, editor, *Proceedings of the Second Workshop on Information Hiding*, volume 1525 of *Lecture Notes in Computer Science*, pages 344–254, April 1998. 5.1.1
- [132] Roberto Zunino and Pierpaolo Degano. A note on the perfect encryption assumption in a process calculus. In Igor Walukiewicz, editor, *Proceedings of the 7th International Conference on Foundations of Software Science and Computation Structures*, volume 2987 of *Lecture Notes in Computer Science*, pages 514–528, April 2004. 3.3



## A Paper Submitted for Publication

This appendix contains a paper submitted to the 10th European Symposium On Research in Computer Security at April 1st, 2005.

The paper was written in a very early stage of the project, meaning the content is not indicative of the quality or factual content of this thesis.

The paper was rejected for publication and appearance at the conference.

# A Framework for Adversary Models

Ole Kasper Olsen and Einar Snekkenes  
ole.olsen@hig.no, einar.snekkenes@hig.no

Norwegian Information Security Laboratory – NISlab  
Department of Computer Science and Media Technology, Gjøvik University College  
P.O. Box 191, 2802 Gjøvik, Norway

**Abstract** The security effectiveness of countermeasures depend on the capabilities and opportunities of the adversary or adversaries. In many cases security claims are made without an explicit adversary model. The lack of a clearly defined adversary model may reduce the value of the analysis results. This paper provides framework for modelling adversaries. It is shown how existing adversary models fit in the framework. The framework can be used to simplify the work of documenting and clarifying assumptions prior to and during security effectiveness analysis.

## 1 Introduction

Information security is largely based around the task of defending an either tangible or intangible object or entity from an attacker—an adversary. Thus, underlying every assessment of security, there is an adversary model present—directly or indirectly.

In this context, a model of the adversary comprises the assumptions which have been made with regards to the specific adversary in question. Often, the security of a measure is based on this set of assumptions. In view of a specific adversary with a specific set of assumptions, a solution may have a high degree of security. In view of a different adversary with a largely different set of assumptions—or even only subtly different assumptions—the same solution may be totally insecure.

It follows that the underlying adversary model is an important aspect of many security measures and research results. Yet, in many cases, assumptions regarding the underlying adversary is scattered and difficult to get an overview of.

## 2 Some Adversary Models

Adversary models come in many different shapes and forms, and to introduce the reader to the concept of an adversary model, this section will present some different examples found in different kinds of literature.

One of the most widespread and most applied adversary model is the Dolev-Yao adversary model [6], used for cryptographic protocol analysis.

The Dolev-Yao model, somewhat inspired by work done by Needham and Schroeder [11], appeared first in 1981. It uses formal protocol algebra to model a protocol and its principals, of which one may be an adversary. The adversary may only perform functions which are supported by the protocol and its primitives, however he is able to send and write possibly arbitrary messages to any principal whenever he chooses. The Dolev-Yao adversary is following a strict minimalistic set of assumptions, which according to the authors facilitates for a less error-prone way to analyse protocols which may be compromised in complex ways than the then-common informal method.

This is in strong contrast to adversary models created for use within penetration testing, for example Wood's treatment of the malicious corporate insider [15], where the adversary is described as close to a realistic person as possible. To achieve this, Wood have to use a highly informal approach and many assumptions which may or may not be true for one specific adversary of this adversary model is enumerated. This is in high contrast to the Dolev-Yao model, where the nature of the adversary is largely abstracted away—it may be human, but equally probably a human-controlled computer—as the Dolev-Yao model pays little or no attention to explicitly stated human strengths or weaknesses.

Other adversary models are based on computational complexity theory (e.g., [2][3][9][16]). This is for example often the case when the strength of encryption algorithms are to be assessed. However, within this area of information security, it is first and foremost assumed that the adversary knows everything about the encryption scheme, except the key or keys used. This echoes Kerckhoffs' second principle [10], which states that exposure of the encryption algorithm to an adversary will not inconvenience the communicating parties. Also, as stated by Shannon [12], no encryption algorithm with a key length shorter than the plaintext being encrypted is secure confronted by an adversary with unlimited time and resources. As having a key as long as the plaintext to be encrypted is extremely unwieldy, Shannon labelled the field of research concerning shorter keys than plaintexts “practical secrecy”, as opposed to the “perfect secrecy” of encryption schemes with equal-length keys and plaintexts.

Today, however computational complexity theory is the weapon of choice, even though the teachings of Kerckhoffs and Shannon form a base. Complexity theory was pioneered for cryptography by Yao [16] and its application facilitates specific bounds with regards to the adversary's computational requirements. This again may be used to find the likelihood of an adversary breaking a certain cipher.

Among the first to apply computational complexity theory directly to encryption schemes were Goldwasser and Micali [9], who define Shannon's notion of “perfect secrecy” for a polynomially bounded adversary by using what they call “semantic security”—a certain form of indistinguishability of messages. This was within public key cryptography. Bellare et al later showed semantic security for a polynomially bounded adversary within symmetric encryption [2].

Another totally different type of adversary model is to be found within the field of anonymity networks, first coined by Chaum in 1981 [5]. The goal of an

anonymity network is to protect information regarding who is talking to whom on a network, and often also the content of the transmitted messages. One specific implementation of anonymity networks is the Onion Routing network [8]. As evidenced by [14], the adversary model in such networks is often a compounded adversary, i.e., consisting of several basic adversary models, as adversaries operating alone can glean little or no information from the anonymity network.

### 3 Related Work

Not much work has been made to attempt a unifying framework for adversary models from all areas of information security.

Wood's description of an insider [15], may be seen as a framework for adversary modelling, especially within the realm of penetration testing. Wood introduces many important attributes detailing adversary assumptions which are important to evaluate in view of setting up a penetration test experiment. However, his penetration testing framework is not directly applicable to other areas of information security.

Other work has also been made to focus on assumptions made with regards to the adversary, as in [4], where a strong focus is set on the explicit and implicit assumptions about the already mentioned Dolev-Yao adversary model. The authors specify a formal framework for distinguishing the assumptions inherent in the model to be able to analyse its strengths and weaknesses.

CORAS [7] is a framework for modelling the risk management process in real systems. In CORAS, UML use case models are merged with so-called misuse case models, to create a better overview of what the system's requirements for use and misuse are.

The technique used in CORAS, is a framework for *threat modelling*, which is integral during the design phase of a system, where system designers and developers need to identify possible attack venues into the system. What sets threat modelling apart from our adversary modelling framework is that threat modelling largely focuses on listing the potential threats certain adversaries may impose on a system, while our framework aims to apply focus on the adversary himself and the ways he may affect the information flow of a system. As with most other schemes for threat modelling, CORAS sees the system from an adversary's point of view, and tries to anticipate his attack goals.

The main goal of our framework however, is to facilitate a better understanding of assumptions already made about an adversary. In that regard, our framework may be complementary to threat modelling in the way that it might help designers to more easily see assumptions they have made with regards to adversaries, and discover similarities between adversaries. This will subsequently make it easier to implement defensive measures, even across different systems, as our framework imposes no limits on the system in use.

## 4 The Framework

The framework we propose will help researchers and security analysts to get an overview of a specific adversary model. Often, the important information about assumptions made with regard to the adversary is scattered or even only implicitly stated in key documentation for security measures or in scientific research reports.

The framework we propose may be used to achieve an initial description of an adversary model of a scenario or real environment. It may also be used as a means to easier make sense of—and realise the potential of—often implicit adversary models found in scientific literature.

We have identified what we believe to be the key attributes in any adversary model. These attributes will simplify the task of gaining an overview of a potentially chaotic area of knowledge, and thus be the backbone of our framework.

The following list enumerates the adversary model attributes involved in the framework, and their hierarchical composition.

1. Principals
2. Channels
3. Protected Asset
4. The Adversarial Setting
  - (a) Adversaries
    - i. Channel Operations
    - ii. Capabilities
    - iii. Resources
  - (b) Intra-Adversary Channels

Attributes 1–3 of the adversary model sets the necessary context for us to successfully be able to model the adversary, which we will do by considering attributes 4a and 4b.

We will now explain the key attributes in our framework by way of an example. Recall the Dolev-Yao adversary model for protocol analysis from section 2. This model is, as mentioned, a model with a fairly small set of assumptions, and quite rigorously described in the literature, and it is still a prevalent model for use within protocol analysis, even 20 years after its inception.

### 4.1 Principals

Any adversary model will refer to a certain set of principal participants in a security system. The principals are the persons or computers interacting in the system faced by a possible attack from an adversary or adversaries. Generally, this means any individual or system/machine with which the adversary may interact in any way.

Given our Dolev-Yao setting, the principals of the adversary model are the communicating parties using the protocol. The principals may be involved in several instances of the protocol, and as such may fill more than one role as defined by the protocol.

At a minimum there are two principals in the Dolev-Yao model, as what is described are two-party protocols for exchange of encrypted messages.

## 4.2 Channels

The channels are what facilitate information flow between principals. Hence, the principals of the adversary model will interact through these channels. The channel over which they interact may be very different, depending on the adversary models and situations in which they apply, however there needs to be a connection between any two or more intercommunicating parties.

The different kinds of channels are characterised by the nature of the channel, the channel's direction and the channel's bandwidth.

Some examples of the nature of channels may be speech and vocal interaction, a physical cable in a network (where a network can be seen as multiple channels between several interconnected principals), sign language or other visual forms of communications and of course various forms of written communication. At a different level we also have the communication channels which are opened on top of underlying channels. On top of a network we may have specific channels defined by their operating protocol, such as email, Web traffic, instant messaging services, and much more—each potentially with their own set of interconnected principals.

The direction of the channel is given by the direction information may flow. An example of this distinction may be found in high-security layered systems, such as those following the Bell-LaPadula model [1]. Here, information may only flow from a lower or equal level of security, often summarised as “no read up” and “no write down”. As a result a channel connecting a low-level security principal with a high-security one will only be one-directional; from the low-level principal to the high-level one.

The bandwidth of a channel may play a role in the modelling of the adversary, although often the channel has for all intents and purposes unlimited bandwidth. Limited bandwidth channels are for example often seen within the field of steganography or in other cases where forms of covert channels are in use.

Returning to our example model, the Dolev-Yao model is a straight forward protocol model, which does not necessarily put any restraints on the type of channel which is being used. It is however clear that [6] is based on a networking scenario. We therefore have bidirectional network channels between involved parties of the protocol, where the channels are provided by the protocol (or the knowledge of the protocol by the participating parties).

## 4.3 Protected Asset

Information security is the defensive discipline of computer science, as security systems are being put in place to protect some asset. The goal of the adversary in a specified model will be to break the protection around this asset in order

to obtain access to it. This asset may be something wildly different depending entirely on the adversary model and scenario. Examples includes anything from confidentiality and integrity of messages and anonymity of communicating parties to the protection of a person or a somehow valuable physical object.

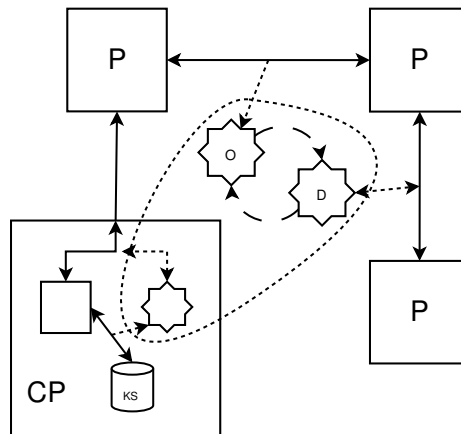
To return to our example model, in [6], the protected asset is the confidentiality of the messages that are being transported over the channels opened by the protocol.

#### 4.4 The Adversarial Setting

We have up until now presented the necessary context in which an adversary may operate. The following sections will focus on the assumptions made with regards to the adversarial setting in which the system operate, given the assumptions made with regards to the context, as described in previous sections.

The adversarial setting comprises one or several adversaries, the channels which connects them, and the adversaries' interface with the system under attack. The individual adversaries' access to the different channels between the principals are defined by a set of possible channel operators.

Figure 1 gives an overview of the adversarial setting a system might face. Solid lines are channels between principals ( $P$ ), dotted lines indicate operations towards the channels (explained below) and the dashed lines are two different intra-adversary channels. The dotted boundary indicates the adversary as a whole. The figure also contains a compromised principal ( $CP$ ), in which a adversary have access to a secret (here it is a keystore ( $KS$ )). The adversaries  $O$  and  $D$  are observing and disrupting adversaries respectively.



**Figure 1.** A generic overview example

**Adversaries** As noted, an adversarial setting may consist of several adversaries. Each of these adversaries have their own set of operations they may perform on the channels between principals, and their own set of capabilities and resources.

**Channel Operations** For each identified channel between principals, the adversary will have a certain set of operations which he may carry out on the channel, or on the messages which are transmitted over the channel.

The following operations are defined:

**Read** By having *read* access to a channel, an adversary will be able to monitor and read all messages which are being transmitted on the channel without exceptions.

**Intercept** The *intercept* operation is defined as the act of being able to block the transmission of a selected message over the channel. The message will then appear to be sent successfully by the sender, but it will never reach the recipient.

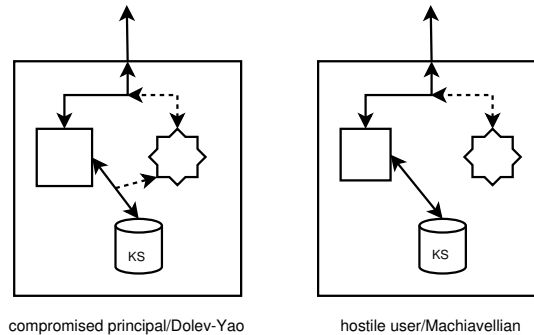
**Write** With full *write* access to the channel, an adversary may introduce his own messages onto channel, meaning he may also replay messages he has *intercepted* or *read*. Often, full-fledged write access is only provided if the adversary is acting as a principal.

Different types of adversaries may be constructed from these basic channel operations. Syverson et al introduce some basic types of adversaries for use in their assessment of security in the Onion Routing anonymity network [14]. These include the “observer”, the “disruptor” and the “compromised COR” (COR is a router in the anonymity network). The “observer” is essentially an adversary with only *read* access one or more channels. The “disruptor” on the other hand has in addition to *read* access, also the ability to *intercept* messages and also *write* new content to the channel. The “compromised COR” is a compromised principal of the anonymity network, and has at least *read* and *write* access to its adjacent channels.

The Dolev-Yao model is explicitly clear on the capability of the adversary to influence the channels over which communication finds place. The adversary may perform all of the operations defined above. It is also implicitly clear that the adversary inherits all secrets known by the adversary. In other words, Dolev and Yao depicts an adversary which is in essence a compromised principal of the network. A slightly less pessimistic version of the Dolev-Yao adversary, is the Machiavellian [13] which is a dishonest principal. It has the same channel operations as the compromised principal of the Dolev-Yao model, however it will not blindly use its own secrets in such a way that it will compromise them. This degree of compromise might be modelled by detailing important channels within the compromised principal, as indicated by figure 2.

**Capabilities** One integral aspect of any adversary model, is the capabilities of the adversary which may dictate what the specific adversary may compute, or in other way deduce.





**Figure 2.** An example of degrees of compromise in a principal

When studying a system to analyse the assumption made with regards to an adversary, the capabilities of the adversary often fall into three different classes of certainty.

**Formal Capabilities** These are concrete and definitive capabilities, the adversary is known to have. For example, in the Dolev-Yao model the adversary’s capabilities are explicitly bound in his actions by the algebraic properties of the protocol in use.

**Probabilistic Capabilities** Capabilities that have well defined probabilities associated with them are here called probabilistic capabilities. This is often used within computational complexity theory, where the upper and lower bounds may be set for the probability that the adversary has a certain capability, given for example a certain amount of computational resources. In literature on such probabilistic adversaries within cryptography, these are often expressed in terms of *notions* of what an adversary may do (e.g. [2][9]).

**Probable Capabilities** Capabilities that the adversary may or may not have, are often seen in adversary models based on human adversaries. Such probable capabilities are prevalent in situations where the assumptions with regards to the adversary may not be made with firm conviction (e.g. [15]).

Capabilities may be many different things. In an anonymity network setting we may assume that an adversary of the “observer” type, will likely have the capability of counting messages and creating a history of observed messages which later may be used for statistical analysis.

If we turn to our Dolev-Yao adversary we see that he is bound by the algebraic properties of the protocol. Among other things, this implies that the only way for an adversary to decrypt a message is to obtain the decryption key and compute  $D_k(E_k(x)) = x$ , where  $x$  is the plaintext message. In other words, this implies that the cryptographic operations are performed “as intended”, or “correctly”.

**Resources** These are objects which the adversary may control or has in his possession. These objects may be anything from exploitable persons to access

to computational hardware (computational ability), but also more intangible resources such as time are essential.

Depending on the required level of abstraction, the various details of an adversary's resources may be defined, and they are often strongly connected to the adversary's capabilities. For example, for the adversary to be capable of monitoring and storing traffic in an anonymity network setting, it will need storage capacity.

For our Dolev-Yao adversary, the resources at his disposition are undefined. He may perform any action within his capabilities, to the extent of his computational ability—whatever that may be. Note that the strength of the Dolev-Yao adversary does not lie in his use of resources, but his capabilities and his operational capabilities on the channels he has access to.

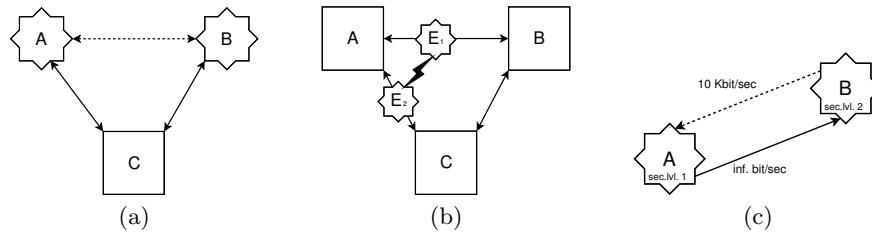
**Intra-Adversary Channels** An adversary may be working alone, or he may be conspiring with other adversaries of potentially different character. In the case of several cooperating adversaries, they may have access to different channels, and combining their areas of influence may produce an adversary who is several orders of magnitude more dangerous than any adversary operating on his own.

For any number of conspiring adversaries, a channel between each of them is required. Let's call this the intra-adversary channel. This special type of channel has the same definition as the channel between the principals presented in section 4.2, and it is characterised by the operations the adversaries may do to it, as specified earlier in this section.

The intra-adversary channel may, as with the channels between principals, appear in many different shapes and forms, and in some cases it may even be the same channel as between principals. One example of this is in the case of compromised principals, where we picture the adversary as being within the principal, and may thus have access to its channel as seen in figure 1.

Figure 3 visualises three different forms of intra-adversary channels, using a somewhat simplified graphical notation than in figure 1. Figure 3a depicts a two compromised principals using the channel between principals as a intra-adversary channel. In figure 3b, two disruptor type adversaries are communicating via a wireless connection. Finally, figure 3c exemplifies a case where two compromised principals are part of a strict Bell-LaPadula like security system, forcing a covert channel with lower bandwidth to enable an intra-adversary channel from higher to lower security levels.

The Dolev-Yao adversary is a formidable one, and in [6] it is not considered whether or not he is conspiring with anyone. The are however nothing in [6] which explicitly states that the Dolev-Yao adversary is an adversary operating strictly on his own, and one may assume that several compromised principals are capable of cooperating. Given a computer networking environment, the channel over which they may cooperate can easily be a channel facilitated via another protocol over computer networks, or they may use the same protocol as between principals.



**Figure 3.** Different forms of intra-adversary channels – (a) A channel between two compromised principals is equal to the channel which must have existed before compromise. (b) Two adversaries of the “disruptor” class have established a wireless network communication channel between themselves. (c) Two compromised principals in a high-security system following the Bell-LaPadula model using a covert channel with relatively low bandwidth to communicate from high to low security.

## 5 Applying the Framework

We will now apply the framework to some adversary models found in various literature from different areas of information security.

### 5.1 Chosen Plaintext Attacks

The chosen plaintext approach to cipher security assessment models an adversary who has access to an encryption oracle, providing him with ciphertexts of the plaintexts he feeds it. The particular adversary model being described here, is based on Bellare et al’s treatment of symmetric encryption [2] where they apply the probabilistic methods of Goldwasser and Micali’s treatment of public key encryption [9].

Bellare et al applies computational complexity theory as briefly discussed in section 2 to symmetric encryption. They give four different notions of security in the vein of Goldwasser and Micali, which we will later see defines some of the capabilities of the adversary.

The oracle is a fairly abstract entity, and may be for example a part of a networked server. Here we assume that the oracle is connected in a network with at least one other principal, else there will not exist any channel which the adversary may exploit.

**Principals** In this scenario, there are two principals; the oracle and a user of the oracle which will query the oracle to obtain ciphertexts.

**Channels** There must be a bidirectional networking channel between the oracle and the other principal.

**Protected Asset** The ability of the encryption scheme under attack is the protected asset in this case, and thus the confidentiality of everyone using the particular encryption scheme.

**Channel Operations** The adversary will have read and write access to the channel between himself and the oracle.

**Capabilities** The adversary is able to conduct statistical attacks on ciphertexts, based on his knowledge of the original language used in the messages, and the plaintexts he has managed to get encrypted via the oracle. He may encrypt as many plaintexts as he likes, at a computational cost.

In the applications of probabilistic encryption however, there are no categorical capabilities. Bellare et al defines four notions of security which may be seen as capabilities, and the underlying probability that the adversary actually inhabits these capabilities. The possible capabilities the adversary may have, are the ability to

- see the difference between a randomly encrypted string from the oracle and one the adversary gives it
- see the difference between two strings he provides the oracle, upon where the oracle returns one of them
- recognise one encrypted string out of several previously and carefully selected plaintext strings provided to the oracle
- learn more information about the plaintext after a ciphertext is obtained, than what was possible before the ciphertext was obtained

**Resources** The assessment of strength of the encryption algorithm is based on a function of the adversary's computational resources, hence there are no definitive limit to the adversary's computational power other than what is realistic that any given adversary may possess. Using computational complexity theory as in [2], it is possible to create bounds on the security of a scheme as functions of the adversary's computational resources. Within these computations, the adversary's resources are given as the number of queries an adversary does to the oracle and the amount of ciphertext seen by the adversary as it is travelling over the channel, along with the running time of the adversary. Given these resources, it is possible to calculate how probable it is that the adversary will mount a successful attack against the cryptographic algorithm in use.

**Intra-Adversary Channels** This is a very theoretical model, and while the resources needed to break the cipher would favour a massively compounded adversary, there are no mention of this. However, even though the model only sees the theoretical adversary as a single one, it may be a compounded one with aggregated resources. As such, no assumptions are made with regards to intra-adversary channels.

## 5.2 Anonymity Networks – Onion Routing

As previously stated in section 2, the goals of anonymity networks are to ensure anonymity with regards to who is communicating with whom, and also what is being communicated. Most practicable anonymity networks are based on Chaum's ideas from 1981 [5]. A network consists of several routers, or mixes. When one party wishes to send a message, a path through the mixes are set up, and the package is repeatedly encrypted in such a way that each mix will be able to remove one layer of encryption, and thereby be able to determine the next mix in the chain towards the destination. Chaum's original mixes was

able to retain the package for a considerable amount of time before passing it on, to severely complicate traffic analysis. This works fine for services such as email where there are no low-latency requirements. For Web traffic and other low-latency protocols however, packets need to be redirected in real time. One mix-net which meets low-latency requirements is the Onion Routing network [8][14], which will be the subject of our adversary framework here.

**Principals** The principals of anonymity network adversary models are those wanting to communicate anonymously (the users of the anonymity network), onion proxies (which sets up the route through the network) and the involved mixes in the network.

**Channels** The channels in the mix network are the network connections between all interconnected mixes and connections between the users of the anonymity service and their local Onion proxies. The channels are all bidirectional and of a nature specified by the Onion Network protocol.

**Protected Asset** The protected asset of the anonymity network are usually twofold. Firstly the anonymity of two communicating parties are protected from prying eyes and traffic analysis methods. Secondly, anonymity networks may also protect the anonymity between two communicating parties from each other.

In addition many networks, such as the Onion Routing network provide confidentiality of the transmitted messages.

**Channel Operations** [14] specifies four different types of adversaries, some of which were mentioned in section 4.4. The “compromised COR” and “hostile user” adversary are two types of principals, although with different functions in the network, and as such may be labelled compromised principals. Both may read and write to the channel(s) adjacent to it. The “disruptor” may use read, write and intercept operations to disrupt the traffic flow of the network. Lastly, the “observer” adversary may only read the channels available to him.

**Capabilities** Capabilities are mainly focused on gleaning information from transported messages and the general message flow of the network. As for the observers, they need the capability of counting messages and storing statistics of the traffic flow or message histories. The disruptors may have the capability of withholding messages indefinitely. The compromised CORs inherit the CORs ability to decrypt messages destined to it. The adversary or adversaries are not assumed to be able to break the layers of encryption which protects the identity of the recipient and sender.

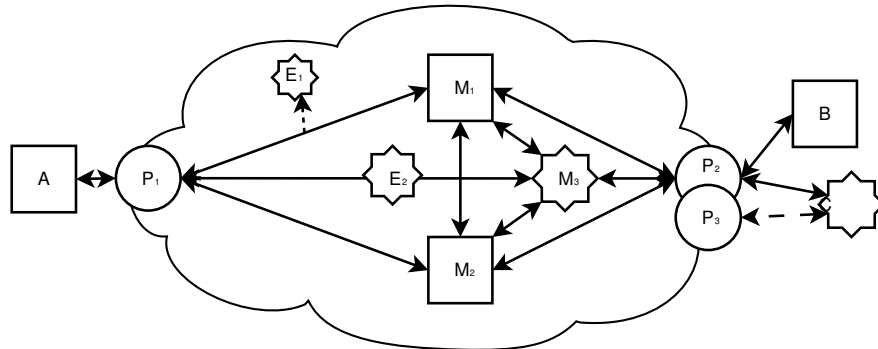
**Resources** No bounds are being laid on the adversary’s computational resources.

**Intra-Adversary Channels** Cooperating adversaries are considered the only truly effective type of adversary in an anonymity network. For example, a global compromised COR adversary is defined when every mix of a mix-net is compromised. Given such an adversary, it will be able to totally break the anonymity traits of the network. Other configurations of cooperating compromised mixes may also be able to learn very much about who is communicating with whom, and even compromised mixes paired with observers

may be able to learn important information, although somewhat dependent on the implementation of the anonymity network. A single compromised COR will only be able to gain information about where messages come from and where they are destined to next.

Given that the Onion Routing network is functioning via a protocol on top of the Internet (TCP/IP), one may assume that the channels between adversaries are made up of the same network connections, only using a different protocol. It is not detailed in which manner the adversaries are cooperating, but it is assumed that the adversaries may communicate in real time or near real time (with low latency).

Figure 4 shows an example of a anonymity network consisting of tree mixes ( $M$ ), of which one is subverted by an adversary and has become a compromised mix. The network has three users ( $A$ ,  $B$  and  $C$ ), of which one is a hostile user.  $B$  and  $C$  are on the same local area network, having two proxies to interface with the Internet; one Onion proxy ( $P_2$ ) and one normal proxy ( $P_3$ ). Other adversaries present are one observer ( $E_1$ ) and one disruptor ( $E_2$ ). The adversaries present may communicate via standard communication protocol over the Internet, indicated by the cloud. Solid lines indicate channels between principals, dashed lines indicate intra-adversary channels and the dotted lines indicate channel operations.



**Figure 4.** Anonymity network adversary model overview.

### 5.3 The Insider Threat

The insider model is a model of a scenario where a knowledgeable insider for some reason turns on the company in which he works. While this adversary may be extremely diverse, and exploit connections and influences he has on an enormous amount of company assets and employees, we will here only focus on the adversary model as described by Wood in [15]. Wood's adversary is a disgruntled

employee who uses his influence and knowledge to attack his company's computer systems, where system is explicitly defined as "the overall network within the scope of some relevant management domain."

**Principals** The involved principals in this insider threat scenario are many. The most important principals are the insider's coworkers or people in other ways related to the company, who knowingly or unknowingly may be a part of the insider's scheme. Additionally, it is in this case natural to consider servers and other computers in the company information infrastructure as principals of the adversary model.

**Channels** The available channels the insider may manipulate are many and diverse, and they depend on the position of the insider within the company and his goals.

One of the most important ones are the channel between the adversary and the internal system in which the target resides. The adversary has either a network connection between himself and the system, or a physical one (direct channel to system, or channel to a privileged person). The adversary may have to use this channel to create new channels into the system until a channel directly into the target within the system is opened. The ultimate goal must be to open a networking channel between the adversary and the target system.

**Protected Asset** The protected asset of this model, is the target within the system under attack. This target can be perceived to be a server containing important company information, or it may be some other important company network infrastructure which the adversary may use for planting logical bombs or similar. Even though the physical target of this adversary may be a computer system, his ultimate goal, and thus the protected asset, are for example monetary gains by selling confidential company information, or subverting or discrediting the company by breaking the integrity of data.

**Channel Operations** The insider may be seen as a hostile user or a Machiavellian adversary. He is an adversarial principal, and reluctant to take risks and part with trails of incriminating evidence. The operations the adversary may do on the network channels are described by his system privileges. Initially, the adversary may only have access to channels within limited parts of the system. However, by applying his full read and write access to these channels, he may compromise other principals (computers in the system) and gain access to channels leading to the target itself within the system. The same applies to physical channels. By "writing" to a physical channel between himself and another human principal, he may coerce or otherwise fool the principal into becoming a compromised principal which he to some extent may control. If the subverted principal is hard to exploit, this may be seen as a bandwidth-limited channel.

**Capabilities** The capabilities of the adversary highly depends on his skills. At the very least, this adversary is able to gather intelligence without arising any suspicion, based on his familiarity with the target. The adversary *may* be a local domain expert on the target, meaning he *may* do anything within

the capabilities of his tools. What the adversary is capable of may also be fuelled by his motive, which may be profit, change-provocation, company subversion or some other personal motive (i.e., hate and revenge).

**Resources** The insider is assumed to have access to most resources within the target system (coworkers, software and hardware), but not necessarily all. As for computational resources, he will have access to most of the company's computers and other hardware.

**Inter-Adversary Channels** The insider is by his own a very capable adversary, however as mentioned he may not have direct access to all necessary channels. To access a certain channel, he may as previously noted fool someone, or he will need to compel colleagues into working with him as an accomplice. He will however, do this only when it is absolutely necessary, as he is very risk averse. So, usually the adversary will work alone.

One very likely accomplice however, is the external employer who may have paid the insider to attack the target.

## 6 Conclusions and Further Work

We have introduced a framework for adversary modelling, and shown the framework in use.

Further work on this subject will be focus on applying the framework to more cases and adversary models, and in the process adapt and refine it.

Being able to use a framework to present adversary models in succinct ways, will facilitate a good basis for the construction of a taxonomy of adversary models used within the field of information security today.

## References

1. D.E. Bell and L.J. LaPadula. Secure computer system: Unified exposition and multics interpretation. Technical Report MTR-2997, The MITRE Corporation, May 1976.
2. Mihir Bellare, Anand Desai, Eron Jorjipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *Proceedings of 38th Annual Symposium of Foundations of Computer Science*, pages 394–403, October 1997.
3. Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13(4):850–864, November 1984.
4. Ilario Cervesato, Nancy A. Durgin, Patrick D. Lincoln, John C. Mitchell, and Andre Scedrov. A meta-notation for protocol analysis. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*, pages 55–69, Mordano, Italy, June 1999.
5. David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.
6. Danny Dolev and Andrew C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, March 1983.



7. Rune Fredriksen, Monica Kristiansen, Bjørn Axel Gran, Ketil Stølen, Tom Arthur Opperud, and Theo Dimitrakos. The CORAS framework for a model-based risk management process. In S. Anderson, S. Bologna, and M. Felici, editors, *Proceedings of the 21st International Conference on Computer Safety, Reliability and Security*, volume 2434 of *LNCS*, pages 94–105. Springer-Verlag, September 2002.
8. David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. Hiding router information. In R. Anderson, editor, *Proceedings of Information Hiding: First International Workshop*, volume 1174 of *LNCS*, pages 137–150. Springer-Verlag, May 1996.
9. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
10. Auguste Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, IX:5–38, January 1883. <http://www.petitcolas.net/fabien/kerckhoffs/>.
11. Roger M. Needham and Michael D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.
12. Claude E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, 1949.
13. Paul Syverson, Catherine Meadows, and Iliano Cervesato. Dolev-Yao is no better than Machiavelli. In *Proceedings of First Workshop on Issues in the Theory of Security*, 2000.
14. Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an analysis of onion routing security. In Hannes Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *LNCS*, pages 96–114, July 2001.
15. Bradley Wood. An insider threat model for adversary simulation, 2000.
16. Andrew C. Yao. Theory and applications of trapdoor functions. In *Proceedings of 23rd Annual Symposium on Foundations of Computer Science*, 1982.