Security in Critical Information Infrastructures

by

Martin Gilje Jaatun

A dissertation submitted in partial satisfaction of
the requirements for the degree
DOCTOR PHILOSOPHIAE (Dr. Philos.)

University of
Stavanger

Faculty of Science and Technology
Department of Electrical Engineering and Computer Science
July 2015

University of Stavanger
N-4036 Stavanger
NORWAY
`www.uis.no`

ISBN 978-82-7644-604-3
ISSN 1890-1387

PhD Thesis UiS no. 254

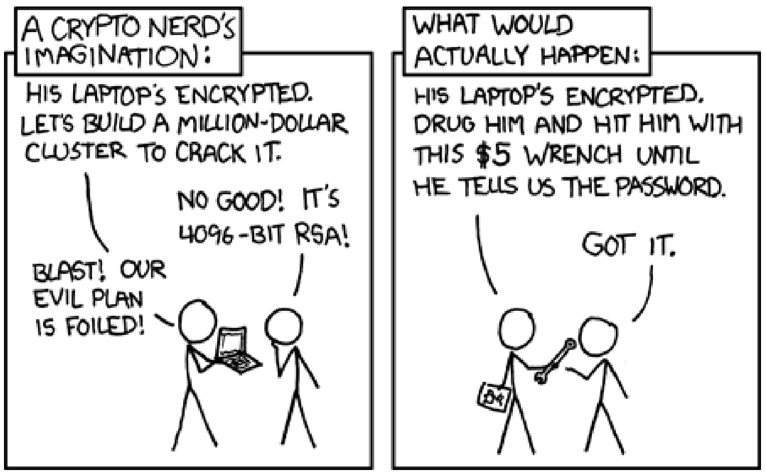
A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

Illustration by Randall Munroe.
Reproduced with permission from http://xkcd.com/538/

Abstract

Information and Communication Technologies (ICT) are permeating the critical infrastructures that our modern society relies on, and ICT security is becoming increasingly important to all aspects of our lives. This dissertation presents security solutions from domains such as telecommunication, oil & gas, and aviation, and shows by contrasting with information about attacks and the threat of malware, that whereas a security architecture is necessary for all security solutions, software security is becoming vital to *all* types of development, not only security software.

This dissertation addresses the following research questions:

(i) How can handovers between wireless access points be performed securely without incurring delays that affects VoIP user experience?
The secure handover scheme using Kerberos tickets (Paper 2, Paper 3) allows for handover between 802.11 access points with a delay between 10 and 40 ms, also when the mobile station has never encountered the destination access point before.

(ii) How can security functions be organized in a critical infrastructure network?
The dissertation presents a novel, simple, yet elegant scheme for asset identification as part of the requirements elicitation process (Paper 4), and contrasts this with a requirements process in the aviation domain (Paper 5). Furthermore, two case studies demonstrating security architectures in two different domains are presented (Paper 1, Paper 7). Finally, the difficulty of measuring the security of software is discussed, emphasizing the importance of ensuring that the software engineering process includes secure software engineering practices, and that the level of adoption of such processes can be used to measure the secure software engineering maturity of an organization (Paper 6).

(iii) How can a system survive a successful attack without alerting the attacker?
The ISH system (Paper 9) represents a novel protection scheme where compromise of a component can be detected by multiple mechanisms, and the compromised component is removed from the system by being promoted to honeypot; achieving the dual goal of both isolating the attacker and providing intelligence on the attack to the defenders.

(iv) How can a successful attack on a critical infrastructure system be mitigated by incident handling?
The IRMA method (Paper 8) is based on sustained interaction with the Norwegian oil & gas industry, and is tailored to this domain with a particular emphasis on learning from security incidents to improve handling of future incidents.

(v) What is the threat posed by malware currently not detectable by signature-based anti-malware systems?
The experiment reported in this dissertation (Paper 10) found that five computers

with the latest version of various anti-malware software were infected by a total of 124 unique malware samples after two weeks of agressive internet activity.

The dissertation discusses the challenges faced by security practitioners when convincing customers and other stakeholders of the need for security, and highlights the importance of making explicit tradeoffs between security on one hand, and cost, functionality and user-friendliness on the other.

Acknowledgements

I am grateful for the support of my wife Ellen and my two sons Jakob and Lars, who never knew I was working on this dissertation during the year we spent in Edinburgh, Scotland. Thanks to my parents, Marit and Harald, who brought me up in a home where books were plentiful and scholarship was encouraged.

I am also grateful to ScotRail for providing an ideal working environment on the journey from Edinburgh to Glasgow; and to Dr. James Irvine of University of Strathclyde who provided me with an office and a stimulating work environment during my year in Scotland.

Thanks to my colleagues in the Information Security Group at SINTEF ICT, and in particular to my frequent co-authors in the enclosed papers, Inger Anne Tøndel and Maria Line. Also thanks to my former colleagues Jostein Jensen and Åsmund Nyre, whose PhD work I co-supervised – this has been a major motivation for me to write this dissertation.

Martin Gilje Jaatun, July 2015

Contents

List of Papers ix

1 Introduction **1**

2 Background **3**

1 Kerberos 3
2 Integrated Operations 7
3 System Safety 7
4 Malware 8

3 CII Security **11**

1 Research Questions 11
2 Open Broadband Networks 12
3 Software Security 15
4 Safety-Critical Environments 18
5 Survivability and Malware 20

4 Discussion **25**

1 We Don't Need Security 25
2 Everybody Wants Security 26
3 We Cannot Afford Security 27
4 Security is a Process 28
5 What is a Critical Information Infrastructure? 28
6 Empirical Research Contributions 29
7 Research Questions Revisited 30

5 Conclusion and Further Work **33**

Paper 1: A Security Architecture for an Open Broadband Access Network **35**

Paper 2: Secure Fast Handover in an Open Broadband Access Network using Kerberos-style Tickets **55**

Paper 3: Extending 3G/WiMAX Networks and Services through Residential Access Capacity **71**

Paper 4: Covering Your Assets in Software Engineering **83**

Paper 5: Sink or SWIM 95

Paper 6: Hunting for Aardvarks: Can Information Security be Measured? 107

Paper 7: Secure remote access to autonomous safety systems: A good practice approach 119

Paper 8: A framework for incident response management in the petroleum industry 139

Paper 9: Survival by Deception 155

Paper 10: Fools Download Where Angels Fear to Tread 173

List of Papers

The following papers are included in this thesis:

- **Paper 1**

  A Security Architecture for an Open Broadband Access Network
  M. G. Jaatun, I. A. Tøndel, M. B. Line (neé Dahl), T. J. Wilke)
  Published in Proceedings of the 10th Nordic Conference on Secure IT Systems (NordSec 2005), Tartu, Estonia

- **Paper 2**

  Secure Fast Handover in an Open Broadband Access Network using Kerberos-style Tickets
  M. G. Jaatun, I. A. Tøndel, F. Paint, T.H. Johannessen, J. C. Francis, C. Duranton
  Published in Security and Privacy in Dynamic Environments, IFIP International Federation for Information Processing Volume 201 (IFIP/SEC 2006)

- **Paper 3**

  Extending 3G/WiMAX Networks and Services through Residential Access Capacity
  F. Panken, G. Hoekstra, D. Barankira, J. C. Francis, R. Schwenderer, O. Grøndalen, M. G. Jaatun
  Published in IEEE Communications Magazine

- **Paper 4**

  Covering Your Assets in Software Engineering
  M. G. Jaatun, I. A. Tøndel
  Published in Proceedngs of Third International Conference on Availability, Reliability and Security (ARES 2008)

- **Paper 5**

  Sink or SWIM
  M. G. Jaatun, T. E. Fægri
  Published in Proceedings of Eighth International Conference on Availability, Reliability and Security (ARES 2013)

- **Paper 6**

  Hunting for Aardvarks: Can Information Security be Measured?
  M. G. Jaatun
  Published in Multidisciplinary Research and Practice for Information Systems, Lecture Notes in Computer Science Volume 7465, 2012

- **Paper 7**

  Secure remote access to autonomous safety systems: A good practice approach
  M. G. Jaatun, M. B. Line, T. O. Grøtan
  Published in the International Journal of Autonomous and Adaptive Communications Systems

- **Paper 8**

  A framework for incident response management in the petroleum industry
  M. G. Jaatun, E. Albrechtsen, M. B. Line, I. A. Tøndel, O. H. Longva
  Published in the International Journal of Critical Infrastructure Protection

- **Paper 9**

  Survival by Deception
  M. G. Jaatun, Å. A. Nyre, J. T. Sørensen
  Published in Computer Safety, Reliability, and Security, Lecture Notes in Computer Science Volume 4680 (Safecomp 2007)

- **Paper 10**

  Fools Download Where Angels Fear to Tread
  M. G. Jaatun, J. Jensen, H. Vegge, F. M. Halvorsen, R. W. Nergård
  Published in IEEE Security & Privacy Magazine

My contributions

The papers herein all contribute to the body of knowledge on how to create secure critical infrastructure systems, through case studies of security architectures, description of novel security mechanisms, approaches to secure software engineering, and evidence on why security is necessary in given situations.

The secure handover scheme using Kerberos tickets (Paper 2, Paper 3) allows for handover between 802.11 access points with a delay between 10 and 40 ms, also when the mobile station has never encountered the destination access point before. The dissertation presents a novel, simple, yet elegant scheme for asset identification as part of the requirements elicitation process (Paper 4), and contrasts this with a requirements process in the aviation domain (Paper 5). Furthermore, two case studies demonstrating security architectures in two different domains are presented (Paper 1, Paper 7).

Further, the difficulty of measuring the security of software is discussed, emphasizing the importance of ensuring that the software engineering process includes secure software engineering practices, and that the level of adoption of such processes can be used to measure the secure software engineering maturity of an organization (Paper 6). The ISH system (Paper 9) represents a novel protection scheme where compromise of a component can be detected by multiple mechanisms, and the compromised component is removed from the system by being promoted to honeypot; achieving the dual goal of both isolating the attacker and providing intelligence on the attack to the defenders. The IRMA method (Paper 8) is based on sustained interaction with the Norwegian oil & gas industry, and is tailored to this domain with a particular emphasis on learning from security incidents to improve handling of future incidents. The experiment reported in this dissertation (Paper 10) demonstrated that the threat of malware cannot be handled by signature-based tools alone, finding that five computers with the latest version of various anti-malware software were infected by a total of 124 unique malware samples after two weeks of aggressive internet activity.

Abbreviations and Terms

AS Authentication Server

ATM Air Traffic Management

BSIMM The Building Security In Maturity Model

C Client

DMZ De-Militarized Zone

FFI Forsvarets Forskningsinstitutt (Norwegian Defence Research Establishment)

FP6 The European Commission's 6th Framework Programme for collaborative research

GPRS General Packet Radio Service

GSM Global System for Mobile Communications, originally Groupe Spécial Mobile

HTML HyperText Markup Language

HTTP HyperText Transfer Protocol

IMSI International Mobile Subscriber Identity

IO Integrated Operations

IRMA Incident Response MAnagement

ISH Increasing Survivability by dynamic deployment of Honeypots

ISP Internet Service Provider

MB Mobility Broker

OECD Organisation for Economic Co-operation and Development

OBAN Open Broadband Access Network

RG Residential Gateway

SAS Safety and Automation System

SeSa Secure Safety

SIL Safety Integrity Level

SINTEF Originally an acronym for "**S**tiftelsen for **IN**dustriell og **TE**knisk **F**orskning ved Norges Tekniske Høgskole", but now redefined as a proper noun (which by convention is spelled in all capital letters)

SIS Safety Instrumented Systems

TGS	Ticket-Granting Server
TGT	Ticket-Granting Ticket
UMTS	Universal Mobile Telecommunications System
V	serVer (in Kerberos)
VoIP	Voice over Internet Protocol
WLAN	Wireless Local Area Network

Chapter 1

Introduction

The last two decades has seen a steady creep of information technology into all kinds of critical infrastructure. Computer communication has evolved into critical infrastructure all by itself, but power generation and distribution, oil & gas production, water distribution and sewage all rely heavily on information technology.

This thesis is composed of a series of papers written over a period of eight years, and documents results from a number of projects executed at SINTEF. The papers are presented thematically, rather than chronologically. Most of my work at SINTEF has been performed in cooperation with my colleagues, students[1] or international collaborators, and I have therefore in most places found it more natural to use the “we” form when a personal pronoun is required.

The first three papers originate from the EU FP6 research project OBAN, which ventured to provide a solution that would allow mobile users to use spare capacity from private WLAN access points as a high-bandwidth alternative to GPRS or UMTS data. SINTEF assumed responsibility of the security activity in the architecture work package of OBAN, and this resulted in a security architecture for the OBAN concept [41] (see page 37) and a novel mechanism for ensuring security for fast handovers between access points [44] (page 57). The final results of the OBAN project were summarised in a journal article [66] (page 73).

The next three papers are broadly in the area of software security, first describing a method to identify assets before eliciting security requirements [40](page 85), then documenting security requirements in an aviation application [29] (page 97), and finally delving into the thorny issue of measurement of (software) security. [25](page 109).

The next paper builds partly on the experience from the security architecture of OBAN, by specifying a security architecture for secure remote access to Safety Instrumented Systems in the oil & gas domain [34] (page 121). This effort is then supported by a framework for incident response management in the same domain [26] (page 141).

The penultimate paper builds on an idea conceived during my time at the Norwegian Defence Research Institute (FFI) [31]. This paper [35] (page 157) describes a prototype system for survivability of web-facing servers through duplication, and additionally a dynamic creation of a honeypot in cases where a single component is compromised

[1]some of which have in turn become my colleagues

by an attacker. The last paper [33] (page 175) reports on an experiment testing the efficacy of anti-malware software against new malware, providing empirical figures with respect to the number of malware instances a user may be exposed to in the window of opportunity between when a new malware instance is created and when updated malware signatures are available.

Chapter 2

Background

This chapter provides some background information useful for understanding the papers summarized in Chapter 3.

1 The Kerberos Network Authentication System

Two of the papers in this thesis rely heavily on the Kerberos authentication system, and I therefore take the opportunity to provide a detailed explanation of Kerberos, based on a series of lectures I held using Stallings' "Cryptography and Network Security" [74] as textbook, and an article I co-authored for Telektronikk [42] .

Kerberos [59] is an authentication protocol developed at Massachusetts Institute of Technology (MIT) originally to protect their own network services. Variants of Kerberos have since then been used as authentication protocol in several products, including Microsoft Windows.

In the past, I've found that Kerberos can be quite difficult to understand for those encountering it for the first time. The following approach has proved fairly successful in explaining the concept.

1.1 An Allegorical Description

Let's cast our imagination back to medieval times, and imagine sending an envoy from the King of Norway to the Emperor of China. This was before RFID passports, and indeed any form of image-based identification, so the envoy needs some other way of proving to the emperor that he is not an impostor. A Kerberos ticket can be viewed as a sort of sealed letter of reference containing a secret password. It is assumed that the recipient (the Emperor) can recognize the King's seal, and that it is not possible to open the envelope without breaking the seal (and the seal cannot be forged). When the King hands the "ticket" over to his envoy, he simultaneously informs him of the secret password within (e.g., "42").

Upon reaching his destination, the bearer of the ticket (i.e., letter of reference) presents the sealed envelope to the Emperor, who breaks the seal and extracts the contents. The Emperor then asks the bearer about the secret inside the letter ("What is the answer

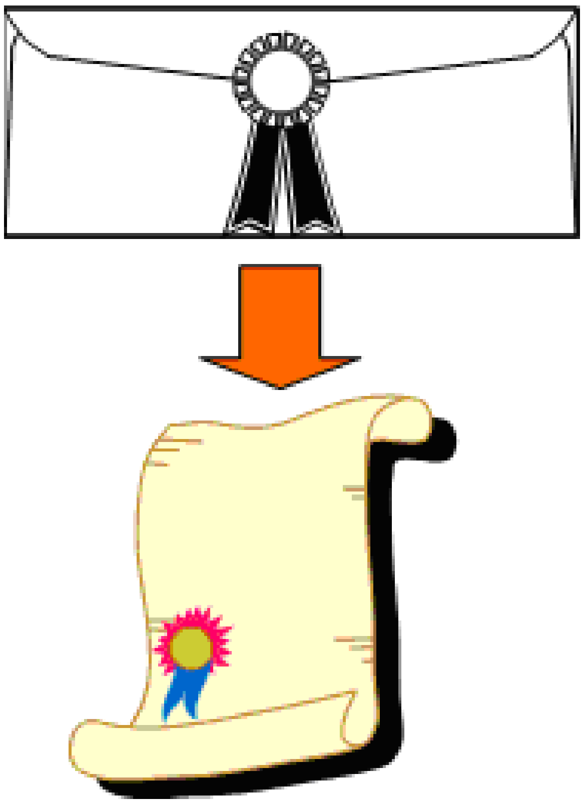

Figure 2.1: A ticket as a sealed letter of reference

to life, the universe, and everything?"). When the envoy responds correctly ("42"), the Emperor can be satisfied that this is indeed the King's trusted servant.

Note that there is one important difference between this allegorical example and a real Kerberos ticket: The letter cannot be opened without breaking the seal, while the Kerberos ticket cannot be "opened" by a third party (or the envoy) at all (we assume that the cryptography is strong enough to make opening "impossible" if one doesn't possess the correct key). If the cryptography can be broken, however, this would have been done without detection, and the whole system would break.

1.2 A Step Closer to the Real World

In a Kerberos ticket, the secret password is a session key (which we in the following will refer to as access key, in order to avoid confusing it with the session key for encrypting the wireless connection), and the seal is realized by encryption with a key shared between the King (or, Authentication Server - AS) and the Emperor (or, Ticket Granting Server - TGS). To complicate matters, there is a third party involved as well, i.e. the server ("V") that provides the specific application service (mail, file transfer, etc.), but for the time being we will ignore this.

The task of the AS is to authenticate the client (done implicitly by issuing data encrypted with the client's password), and issue a Ticket-Granting-Ticket (TGT) that

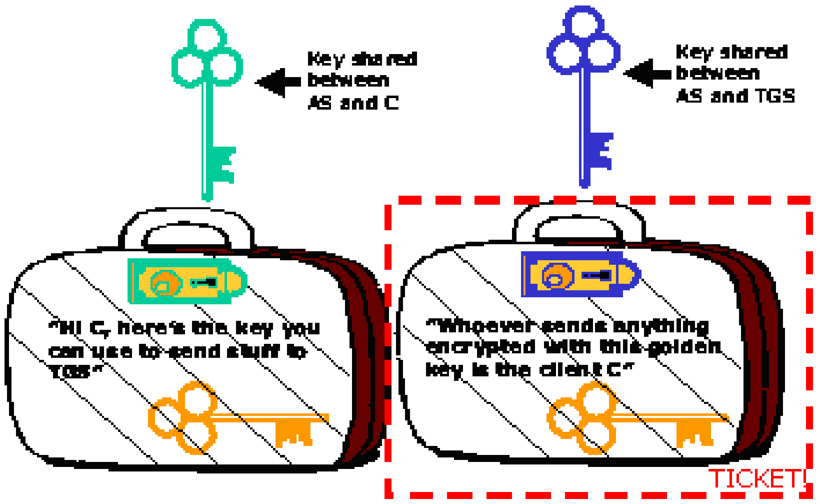
Key shared between AS and C
Key shared between AS and TGS
"Hi C, here's the key you can use to send stuff to TGS"
"Whoever sends anything encrypted with this golden key is the client C"
TICKET!

Figure 2.2: Simplified illustration of ticket and accompanying information

the client will use when authenticating to the TGS. The task of the TGS is to issue a ticket for the specific service that the client wants, as can be seen in Figure 2.4.

When the envoy (or Client - C) initially requests a ticket, it needs a pre-shared secret with the AS, normally in the form of a password. This secret (the green key in Figure 2.2) is used to protect the dynamic access key (chosen by AS) that is also embedded in the ticket. Thus, the ticket in itself is worthless if the bearer cannot prove that it also knows the access key. As illustrated in Figure 2.2, the client C receives two elements from the AS:

- The ticket (which the client cannot decrypt)
- The accompanying information, including (most importantly) the access key (this is encrypted with the shared secret that C and AS possess).

The ticket is used to create a dynamic (indirect) trust relation between C and TGS, based on the fact that there are static (direct) trust relationships in the form of shared secrets between AS and C (green key) and between AS and TGS (blue key). This is illustrated in Figure 2.3.

When finally the service granting server (V) enters the picture, the interactions previously performed toward AS are basically repeated toward TGS, with similar trust relations (except that the direct trust relation between C and AS is replaced with the indirect (dynamic) trust relation between C and TGS). Thus, if we disregard whether trust relations are direct or indirect, we can replace AS by TGS and TGS by V (i.e. serVice granting server) in Figure 2.3. The "full picture" can be seen in Figure 2.4 (adapted from Stallings [74]).

For completeness, Figure 2.5 gives a more detailed explanation of the messages sent from AS to C after initial authentication. This is thus how the client receives the ticket-granting-ticket; the messages when transferring the service-granting-ticket are completely analogous.

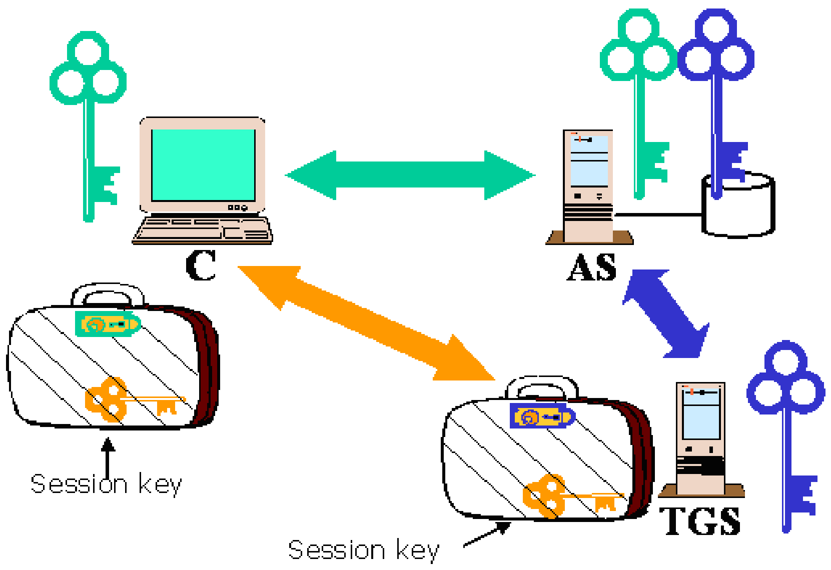

Figure 2.3: Trust relations and shared keys

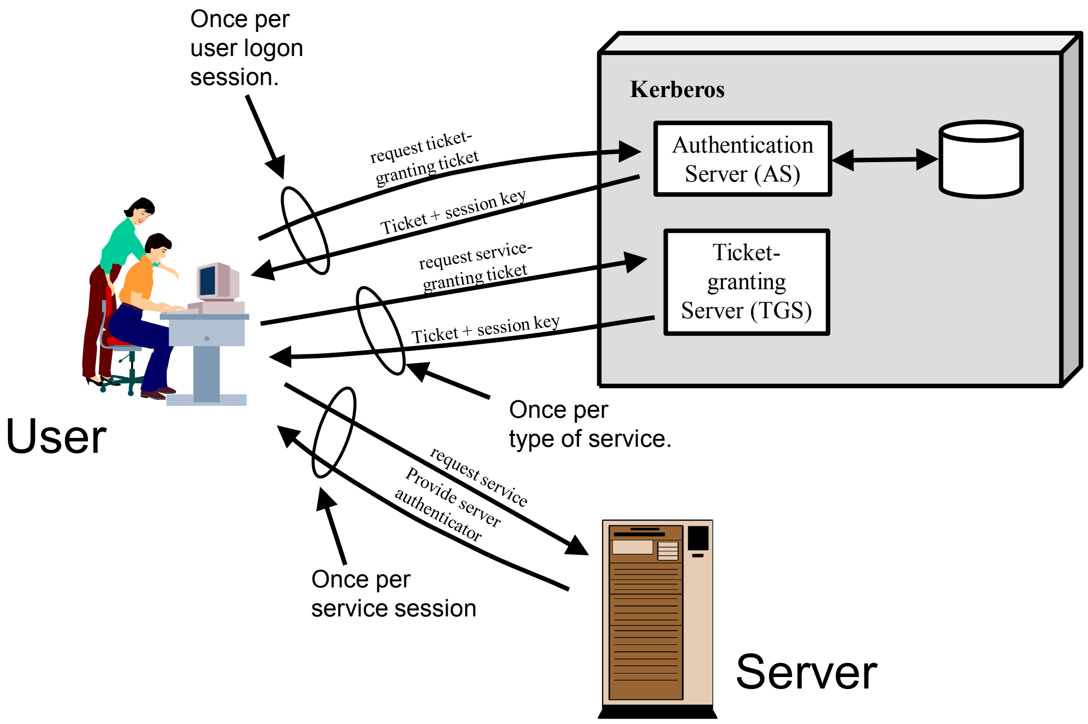
Once per user logon session.
Kerberos
Authentication Server (AS)
Ticket-granting Server (TGS)
request ticket-granting ticket
Ticket + session key
request service-granting ticket
Ticket + session key
Once per type of service.
User
request service
Provide server authenticator
Once per service session
Server

Figure 2.4: Kerberos overview (adapted from Stallings [74])

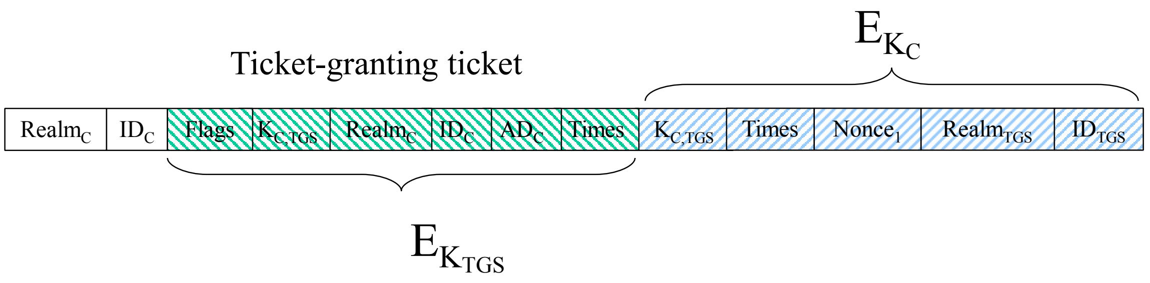

Figure 2.5: Detailed explanation of initial messages between AS and C

2 Integrated Operations

The Norwegian oil industry is responsible for the greater part of the Norwegian gross national product, but since all production facilities are on offshore oil platforms, it is generally dangerous and expensive work. For many years, there has therefore been a push from the industry to increase the level of remote operations, with less reliance on having people flying offshore; if not eliminating humans from the equation, than at least reducing the numbers. This concept has been called "Integrated Operations", and makes heavy use of online collaboration tools, videoconferencing and other ICT.

Integrated operations also implies that previously isolated process control networks increasingly are being interconnected; and operators, contractors, and service companies communicate directly using computer networks. Although the Norwegian Oil Industry Association (OLF) has tried to avoid having Integrated Operations being perceived as a synonym for remote operations[1], it is clear that this is major aspect. In the short term, IO reduces the need for onshore experts to make short-term visits to offshore installations to perform ad-hoc problem solving. This is due to the aforementioned online collaboration tools, allowing many problems to be solved collaboratively, with any necessary manual tasks performed by the offshore personnel already on site.

3 System Safety, Safety Integrity Levels and Safety Instrumented Systems

The terms *Safety* and *Security* refer to related, but quite different concepts. It is therefore interesting to note that in Germanic languages such as Norwegian, there is just a single word ("Sikkerhet") to cover both concepts. This is a constant source of misunderstandings, and can sometimes lead to controversy [52]; two parties may seem to agree that security is very important, but later it will turn out that one of them is only concerned about safety.

There have been many cases in the past [29, 11] where safety experts have tried to

[1]This is likely due to political considerations, since offshore labor unions see remote operations as a potential threat to their paycheck and/or a precursor to offshore job cutbacks.

apply safety methodologies to security problems, with disappointing results. While a safety analysis will try to calculate, e.g., a system-wide probability of failure, it is not possible to calculate a probability of attack in the same manner. Consequently, it is not possible to calculate a ”Mean Time To Attack”.

Safety in process control environments has always been very important, and international standards [68, 4, 5] detail how to design and evaluate safety-related systems (see Figure 2.6). Systems are classified into 4 Safety Integrity Levels (SIL), where a higher SIL indicates a higher reliability of the safety functions. Another way of putting this is that a higher SIL indicates a higher probability that all required safety functions will perform as required when needed. Safety Integrity Levels are thus dependent on the risks that a given system is subject to, and say something about the required risk reductions offered by the safety systems.

The process industry has created its own standard for development and assessment of Safety Instrumented Systems [5]. Even more specifically, the Norwegian Oil Industry Association (formerly OLF, now Norsk Olje & Gass) published a guideline for applying IEC 61508 and 61511 in the Norwegian Petroleum Industry [1]. Safety systems are based on concepts such as “Mean Time to Failure”, allowing individual components to be replaced before they statistically would be expected to fail. An offshore installation represents a substantial investment, and typically the life cycle for process control equipment is significantly longer than for comparable office systems. From a safety perspective, this is not a problem as long as components that may fail are replaced regularly; from a security perspective, however, it often implies that systems with security flaws are kept in service long after the flaws have become common knowledge.

Strangely, whereas information security professionals might feel more secure when using systems that are certified for a high security level, safety professionals can get uneasy when relying on a system certified for a high Safety Integrity Level. The reason for this apparent contradiction is that a system with a high SIL relies more extensively on the safety system, and in one sense less able to tolerate an eventual loss of this system - if things go wrong, they go spectacularly so. This also implies that any information security flaws will have a *higher impact* on a system with a high SIL [21].

4 Malware

Malware can be described as software written with malicious intent[2], or alternatively computer code which exhibits malign or harmful behavior or effects. Examples of malware include viruses, worms, trojans, and rootkits.

The first academic treatment of computer viruses was performed by Cohen [12], and while he also claims to have created the first virus as an experiment in November 1983, we have from other sources that the Elk Cloner virus for the Apple II computer was written in 1982 [51]. The first virus for the IBM PC, the globally propagated Brain computer virus, originated from Pakistan in 1986 [72, 23]. Although the motivation is a

[2]Sometimes, the popular press will use the term “malicious software”, but this would imply that the software is somehow sentient and capable of having independent intentions, which is clearly not the case.

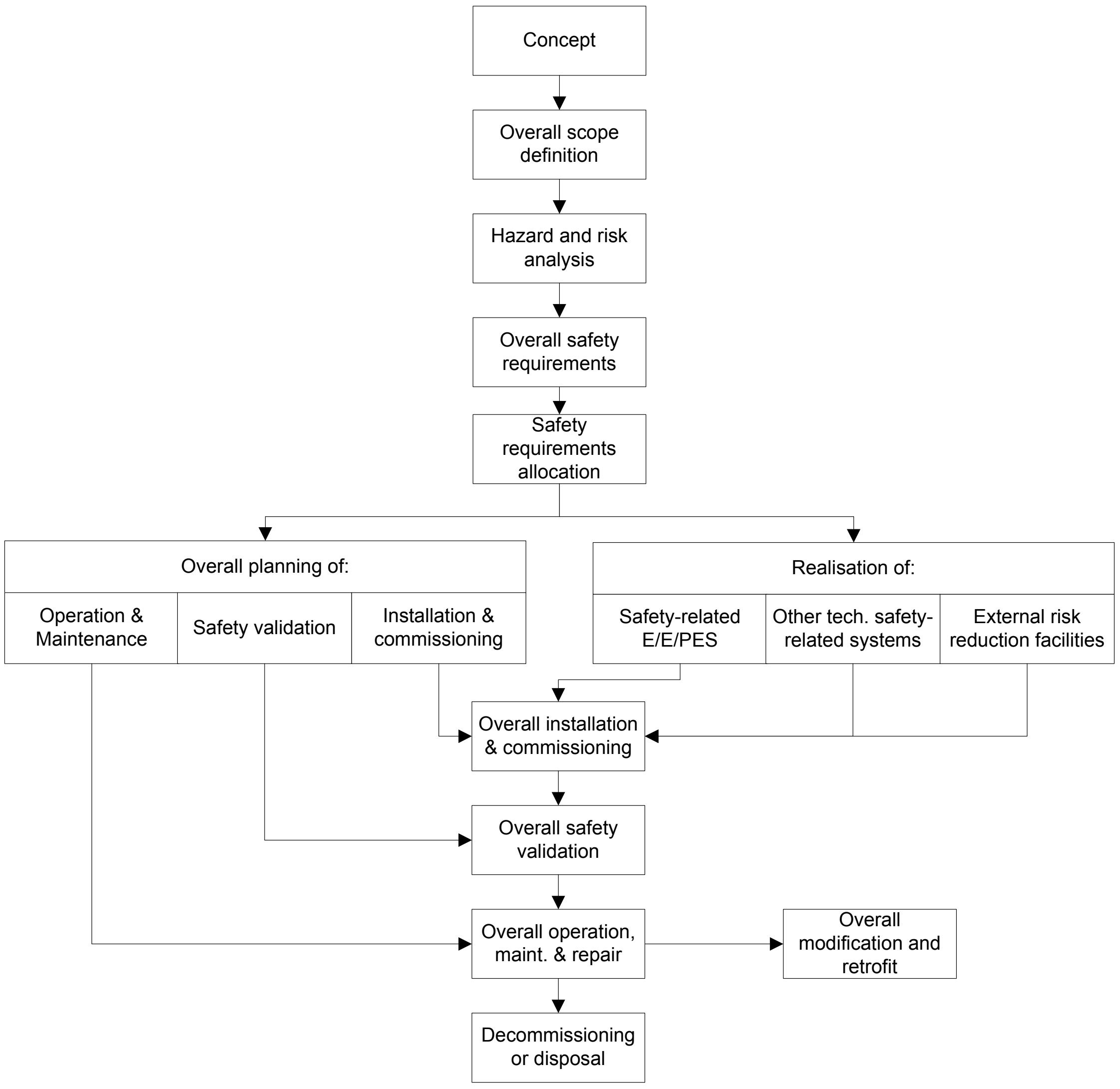
Concept
Overall scope definition
Hazard and risk analysis
Overall safety requirements
Safety requirements allocation
Overall planning of:
Operation & Maintenance
Safety validation
Installation & commissioning
Realisation of:
Safety-related E/E/PES
Other tech. safety-related systems
External risk reduction facilities
Overall installation & commissioning
Overall safety validation
Overall operation, maint. & repair
Overall modification and retrofit
Decommissioning or disposal

Figure 2.6: The safety cycle as illustrated in IEC 61508

bit unclear, Brain was ostensibly developed as a copy-protection measure by a Lahore software firm. The first generation of viruses spread via floppy diskettes, either infecting the boot sectors of connected hard disks or diskettes, or individual programs on the target machine. It is fair to say that software piracy was a major contributing factor to the spreading of these viruses, since games would be copied and distributed manually from person to person, and the viruses would be silently copied along with them.

Whereas viruses first were the province of personal computers such as the IBM PC and the Amiga, computer worms started their life on multiuser platforms such as UNIX [65]. When personal computers began to get connected to the internet in the late 1990-ies, hybrid malware that exhibited both virus and worm characteristics began to appear. While some academics tried to differentiate between viruses and worms by stating that the former required active user interaction, this distinction eventually became less useful, and today most security professionals are content to use the generic term malware.

Many malware instances spread quickly and prolifically because they exploited widely un-patched vulnerabilities in the Microsoft Windows operating system. In particular, the SASSER worm saw such a wide distribution in 2003 that it was impossible to install Windows 2000 on a computer while it was connected to the internet – the computer would be infected and rendered useless before the installation process could even be completed[3].

After the turn of the millennium, there was also a shift in focus of most malware – previously, it seemed that the main purpose of malware was destruction of data or performing more innocent pranks, but now increasingly malware would create back doors on infected systems, enlisting them in so-called bot-nets [80] which in turn could be used to perform distributed denial-of-service (DDoS) attacks against web servers and other systems. Often, this would be part of blackmail schemes, where botnet controllers would extort money from system owners under threat of having their system made unavailable due to DDoS. There have also been examples of so-called *ransomware*, where the malware would encrypt files on the victim's computer, and then demand payment to deliver the encryption key (this form of malware has recently seen an upswing in popularity).

In recent years there have been examples of malware allegedly being used by nation states as part of intelligence or intervention operations. The most famous example is Stuxnet [17], which allegedly caused the physical destruction of nuclear centrifuges at the Iranian Bushehr nuclear power plant, thus hampering the Iranian nuclear weapons program. More recent examples such as Duqu and Flame [18] have been identified as information gatherers focused on strategic technology. The important thing to note here is that malware can be (and has been) used for targeted attacks at critical infrastructure systems, and that this threat cannot be discounted [54].

[3]Personal experience by the author.

Chapter 3

CII Security

This chapter describes the ten papers included in this thesis. They are grouped thematically; papers 1, 2 and 3 are about security in open broadband access networks, papers 4, 5 and 6 are about software security, papers 7 and 8 are about security in offshore process control environments, while papers 9 and 10 are about (resistance to) remote attacks and malware.

1 Research Questions

Unlike a traditional PhD thesis, this work did not start out by first drafting a number of research questions and then subsequently seeking the answers to these questions. However, having co-supervised (formally and informally) a number of PhD students, I realize that this is also generally not true for traditional PhD students – even when the research questions are posed at the onset, they change significantly during the PhD trajectory, and usually end up having to be tailored to the work actually performed, rather than to the work that originally was intended.

With this in mind, we can present the following research questions:

(i) How can handovers between wireless access points be performed securely without incurring delays that affects VoIP user experience?

(ii) How can security functions be organized in a critical infrastructure network?

(iii) How can a system survive a successful attack without alerting the attacker?

(iv) How can a successful attack on a critical infrastructure system be mitigated by incident handling?

(v) What is the threat posed by malware currently not detectable by signature-based anti-malware systems?

The way the papers relate to each other and to the research questions is illustrated in Figure 3.1. The figure is mostly self-explanatory, but the following connections might need some extra explanation:

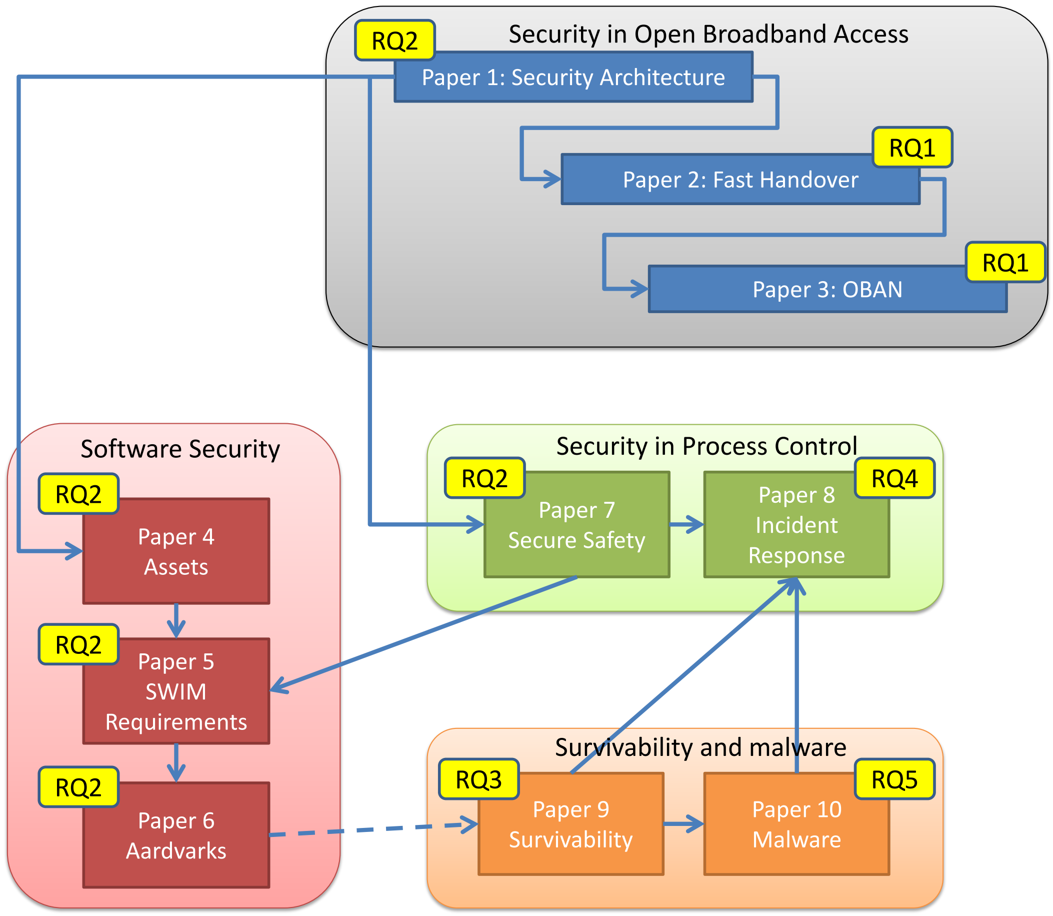

Figure 3.1: Connection between papers

- Paper 1 was our first foray into the area of security requirements, which eventually led us to explore the field of software security and asset identification (paper 4), and security architecture of critical infrastructure systems (paper 7)

- Paper 7 explored the interplay between security and safety, which proved relevant for Paper 5

- Measurement of security (paper 6) is relevant for survivability of a software system (paper 9) (although this link is implicit; hence the dashed line)

2 Open Broadband Networks

Every day, additional households all over the world are getting a broadband connection to the internet; the OECD average currently exceeds 25 broadband connections per 100 inhabitants [64]. These connections are however typically not used with a constant intensity during the course of a day; on weekdays, most residential broadband connections

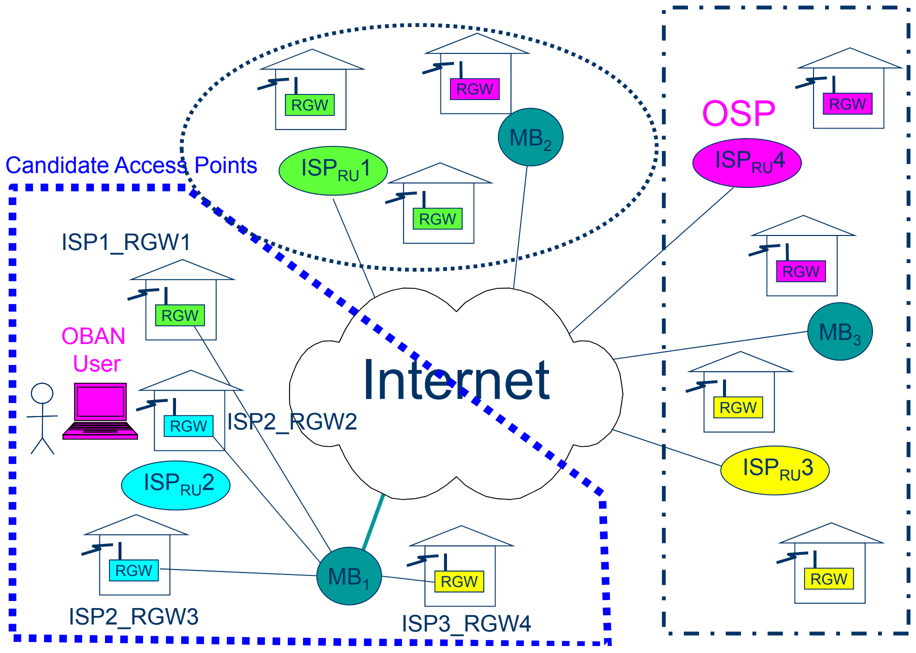

Figure 3.2: Handover candidates per Mobility Broker

are idle when their owners are at work or at school, and the same is true during vacations. This implies that there is a lot of untapped bandwidth[1] that could have been put to good use for others. This was the idea behind the Open Broadband Access Network (OBAN) project (Paper 1-3 [41, 44, 66]), which sought to make excess capacity from personal wireless access points available to passing strangers.

The business case of OBAN was to integrate the personal access points in the roaming hierarchy of mobile telephone operators, allowing subscribers to utilize the high capacity of a wireless access point when available, and hand over to the regular cellular infrastructure when moving out of range. This means tying the use of an access point to a mobile subscription, and also allows for introducing different incentives for the home subscribers for opening up their networks, e.g., reduced subscription cost or increased bandwidth for the same price.

Opening up a private access point to the general public requires a proper security architecture [41]. However, one of the quality-of-service requirements of OBAN was to support seamless handovers for a pedestrian user in urban areas. This proved to be a challenge when performing handovers between two private access points that don't even know about each other, much less have a pre-existing trust relationship. Early trials showed that if roaming users would have to perform an authentication with the provider's back-end systems, the introduced delay would have exceeded the target of 50 ms, which means that, e.g., Voice over IP (VoIP) users would have experienced quality degradation during handover.

[1]Bandwidth is here used in the conventional meaning of "capacity", although some claim it should be reserved for discussions involving a frequency spectrum.

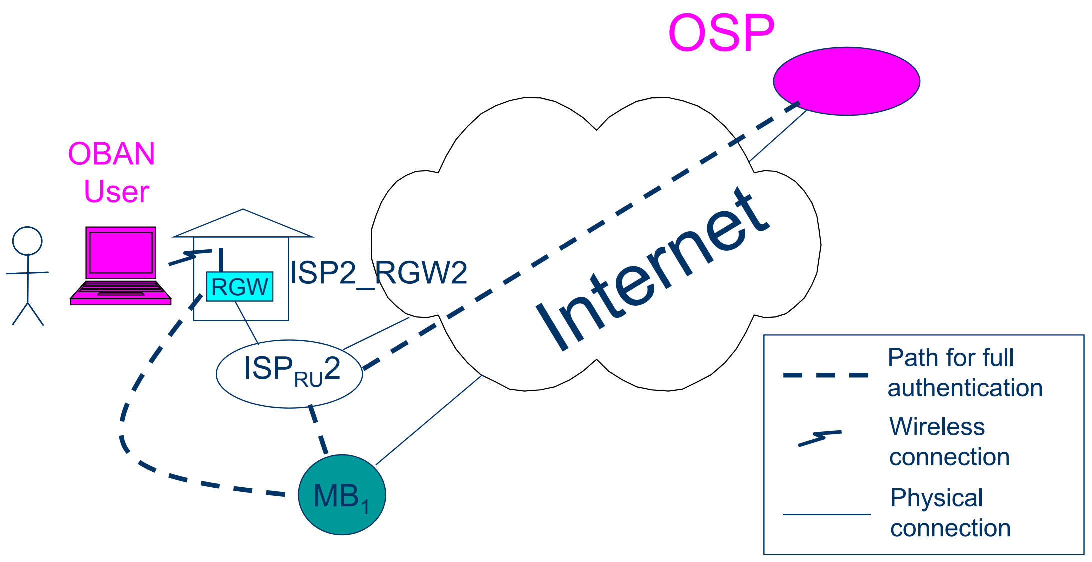

Figure 3.3: Full authentication of mobile user toward home ISP

The solution [44] was to exploit the fact that since all access points would have a trust relationship with the Internet Service Provider (ISP), it should be possible to *pre-authenticate* to all of the surrounding access points (and thus potential handover candidates) before the handover takes place. This requires that the user spends a certain minimum of time at each access point before performing a handover, but since the target group was pedestrian users, this requirement was not deemed to represent a problem. The actual authentication method selected was to use Kerberos tickets which could be evaluated locally on each access point.

However, since in any given residential area there may be more than one ISP, it may be unnecessarily restrictive and inefficient to only enable handovers between access points belonging to the same ISP. In OBAN, it was therefore decided to introduce a new geographically defined entity; the *Mobility Broker* (MB). All participating access points would need to establish a trust relationship with the MB in their area; this would typically be done at time of installation. The MB would in turn have a trust relationship with all participating ISPs.

Using Kerberos-style tickets is feasible since the MB's trust relationship with every access point means that there is a possibility of establishing a shared secret between the MB and any given access point. This shared secret can be the encryption key used to encrypt the "service ticket" which is issued to the mobile user, and which in turn contains the session key that is used to authenticate and establish a secure connection with the mobile user during the handover.

It is important to note that the mobile user will try to acquire Kerberos tickets to all potential handover candidates before it becomes necessary to perform the actual handover. The task of determining which access points are likely candidates is left to the Mobility Broker, and in some cases it might be feasible for the MB to issue the

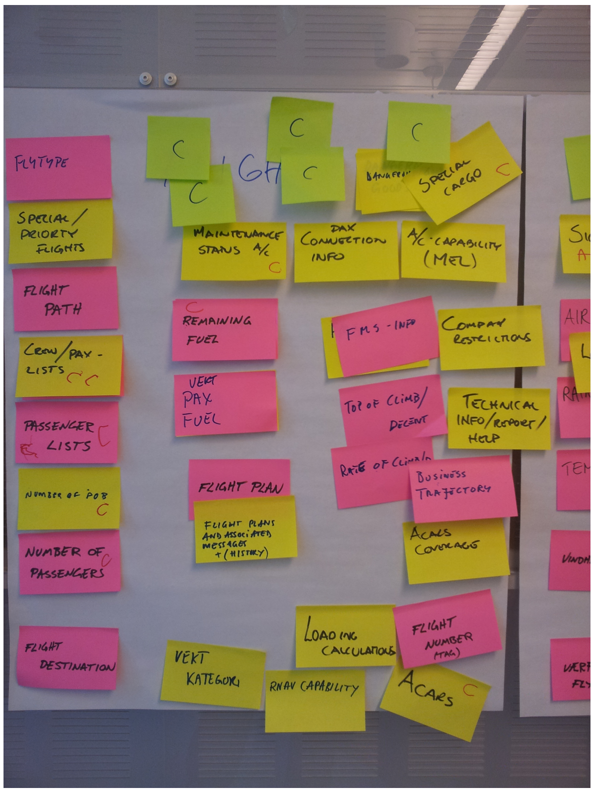

Figure 3.4: Brainstorming with Sticky Notes

mobile user with tickets for *all* the access points in its area. In any case, it is only necessary for the mobile user to perform one full authentication toward its home ISP prior to acquiring handover tickets, as illustrated in Figure 3.3.

3 Software Security

Secure systems are not simply collections of security features, but also require that all the implemented features are secure. This implies that not only does care need to be taken when coding critical infrastructure applications, but in the early stages of development it is important to identify the assets that are to be protected before security requirements are elicited.

We found that identification off assets is something that most secure software methodologies mentioned as an important first step, but none of them provided any information on exactly how this should be done. Inspired by brainstorming techniques as described by Dybå et al. [15], we formalized in Paper 4 [40] a step-by-step process for identifying and prioritizing assets.

First, a group of stakeholders with detailed knowledge of the system to be built (or analyzed) are assembled in a room, led by a moderator. There are no firm limits with respect to the number of attendees, but we found that it works best with 8 stakeholders or less. Technical experts and actual users are generally more useful than managers

Table 3.1: Asset prioritization table

ASSETS	**STAKEHOLDERS' PRIORITY**		
	Focus: protection What is most important to protect from stakeholders' point of view?		**Focus: attacks** What is most interesting or valuable for an attacker?
Description	**System user**	**System owner**	**Attacker**
<asset 1>	<C-? I-? A-?>	<C-? I-? A-?>	<C-? I-? A-?>
<asset 2>	<C-? I-? A-?>	<C-? I-? A-?>	<C-? I-? A-?>
...	...	...	...

in this phase, and the latter should be avoided if they are likely to put a damper on creativity. The stakeholders are then given pads of sticky notes, and asked to write down all assets they can think of in the system at hand. This continues as long as anyone can think of new assets to write down; when activity tapers off, the process leader announces that this phase will continue two minutes more (up to the discretion of the moderator).

The next phase starts with the moderator asking each stakeholder in turn to put (say) five notes up on the wall (e.g., a whiteboard) as illustrated in Figure 3.4. This continues in a round robin fashion until all notes are up on the wall. Stakeholders are encouraged to write new notes during this process, if spurred on by other stakeholders' ideas. They can also group notes that are duplicates or related to each other.

When all the notes are up on the board, the moderator can assist with further grouping if necessary. The stakeholders are then asked to rank the importance of confidentiality, integrity and availability of each asset using a qualitative scale of High-Medium-Low, and to do this from three perspectives: the user (or customer), the owner of the system (or developer), and the attacker. This usually requires some discussion among the stakeholders until consensus is reached. The end result is then filled into a table as exemplified in Table 3.1. The final priority for an asset can be calculated by assigning values to the qualitative scale, and summing up the values across the table. In our original method [40], we assigned the value 1 to "High", 2 to "Medium" and 3 to "Low", and 4 to any values not entered, thus giving the lowest resulting value the highest priority. In later work [32], we reconsidered this, flipping the scale with a 0 for no value, 1 for "Low" etc. We also decided that summing up the CIA values did not make sense, and instead calculated individual priorities for confidentiality, integrity and availability for each asset.

Having identified the assets, the next step is to elicit security requirements to protect these assets. As an example of this, we can take the security requirements we elicited for the aviation System Wide Information Management (SWIM) system (Paper 5

[29]). In this case, the primary assets (information and services) and secondary assets (hardware and infrastructure) were defined in a complex process involving both numerous stakeholders and multiple development organizations.

SWIM is introduced as a replacement to the traditional point-to-point communications in aviation (see Figure 3.5 and Figure 3.6), ensuring that information such as flight trajectory and meteorological data can be shared by a large number of actors including pilots, Air Navigation Service Providers and Airport Operation Centres. SWIM provides a unified interface to aviation information, supporting more efficient Air Traffic Management.

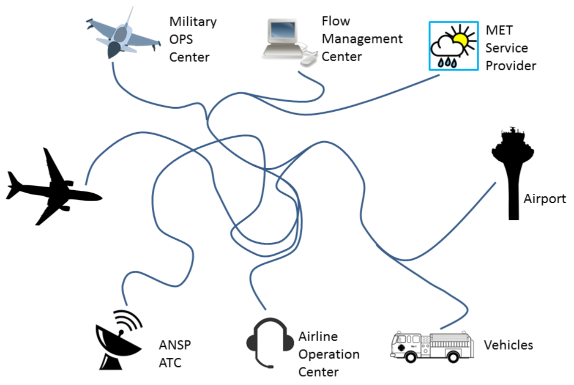

Figure 3.5: Aviation communication before SWIM

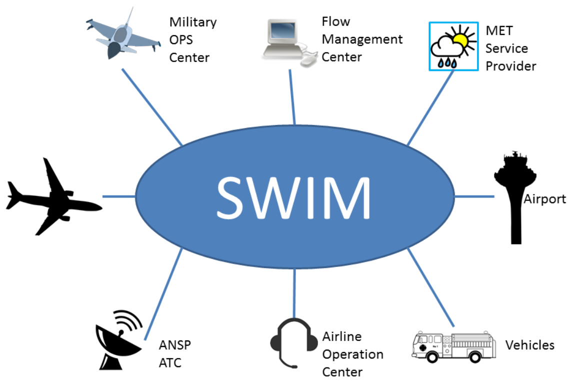

Figure 3.6: Aviation communication after SWIM

Once a piece of software has been written, it is very difficult (if not impossible) to *measure how secure* it is, and equally difficult to compare two programs to determine whether one is more secure than the other. This is the problem tackled by Paper 6 [40], which looks at various approaches to software security metrics before concluding that the most promising currently is the Building Security In Maturity Model (BSIMM). Instead of trying to measure the security of the software that is developed, BSIMM

Table 3.2: The BSIMM Software Security Framework

Governance	**Intelligence**	**SSDL Touchpoints**	**Deployment**
Strategy and Metrics	Attack Models	Architecture Analysis	Penetration Testing
Compliance and Policy	Security Features and Design	Code Review	Software Environment
Training	Standards and Requirements	Security Testing	Configuration Management and Vulnerability Management

measures second-order effects, i.e., which activities a development organization performs when developing (secure) software.

BSIMM presents a software security framework consisting of 12 *practices* organized in four *domains* (see Table 3.2). Each practice contains a number of *activities* categorized in three maturity levels, where the assumption is that the activities on the first level should be covered before activities on the higher levels should be approached. However, BSIMM claims to be *descriptive* rather than *prescriptive*, implying that it should be used for measurement rather than a standard to aspire to. Having said this, it is nonetheless evident that if organizations perceived as software security leaders are implementing a set of activities from the BSIMM Software Security Framework, this will be a strong incentive for other organizations to do the same.

Thus far, BSIMM has mainly been used as measurement tool by Digital (the company that initiated BSIMM), and as a self-assessment tool. It remains to be seen if it can also be used as a research tool, or if the personal qualities of Cigital's "BSIMM auditors" are too specialized to be reproduced by random scientists.

4 Security in Safety-Critical Environments

The Norwegian offshore oil & gas industry is embracing the Integrated Operations (IO) concept, which means that it experiences an increased reliance on information and communications technology. This in turn means that greater efforts have to be expended on building more secure systems, but also that contingency measures such as frameworks for information security incident response management have to be developed.

4.1 Secure Safety

Safety Instrumented Systems (SIS) are tasked with ensuring the safety of process control environments, and it is therefore imperative that an outside attacker be prevented from interfering with them. The introduction of Integrated Operations (IO) on the Norwegian continental shelf implies that process control networks increasingly are being interconnected.

The Secure Safety (SeSa) project (Paper 7 [34]) ventured to specify a layered security architecture based on the principle of defense in depth, taking into account the various safety levels with associated required integrity. It should be noted here that in the context of safety, integrity of systems is the over-arching concern; confidentiality may be needed for other reasons, but was considered out of scope for SeSa. Furthermore, availability of service is normally not a safety concern, since a system that is doing nothing generally does not pose a safety hazard. The obvious exception to this rule is the SIS itself, which if unavailable would not be able to intervene when necessary.

One challenge with the way offshore oil platforms are operated in the North Sea is that the operating organization typically isn't self-sufficient, meaning that specific equipment is owned and operated by external contractors. This also implies that these external contractors sometimes need remote access to their equipment for maintenance purposes. The security zones illustrated in Figure 3.7 was our approach to handle this. As the figure illustrates, contractors accessing the process control network from a secure location would be allowed in, but other connection attempts would be rejected. No direct access would be allowed from the outside to SAS or SIS, but we specified an "information diode" functionality that could mirror content from the SAS history server to a duplicate server in the inner DMZ. This would allow, e.g., contractors read-only access to information for monitoring purposes.

The SeSa method, in addition to specifying a recommended network architecture, also provides a scheme for determining whether a solution will achieve the desired Safety Integrity Level (SIL).

4.2 Incident Response Management

Acknowledging that 100% security in practice is unattainable [71], it is prudent to plan and prepare for situations where information security incidents have occurred and need to be handled.

Since process control systems historically were isolated networks with no external connections, general consensus in the process control community seemed to be that this was a non-issue[2], and some have reportedly claimed that they do not have ICT systems – "we only have programmable logic.". However, during the IRMA project (Paper 8 [26]) it quickly became clear that incidents of various stripe did and do happen – in group interviews and workshop sessions, representatives from the same company would relate different incidents that the others were not aware of. We also experienced that routines or procedures that were identified as missing during one workshop suddenly would be available the next – we interpret this as an example of a learning organization.

[2]Although the Stuxnet example shows that even an air gap is no guarantee against outside attack.

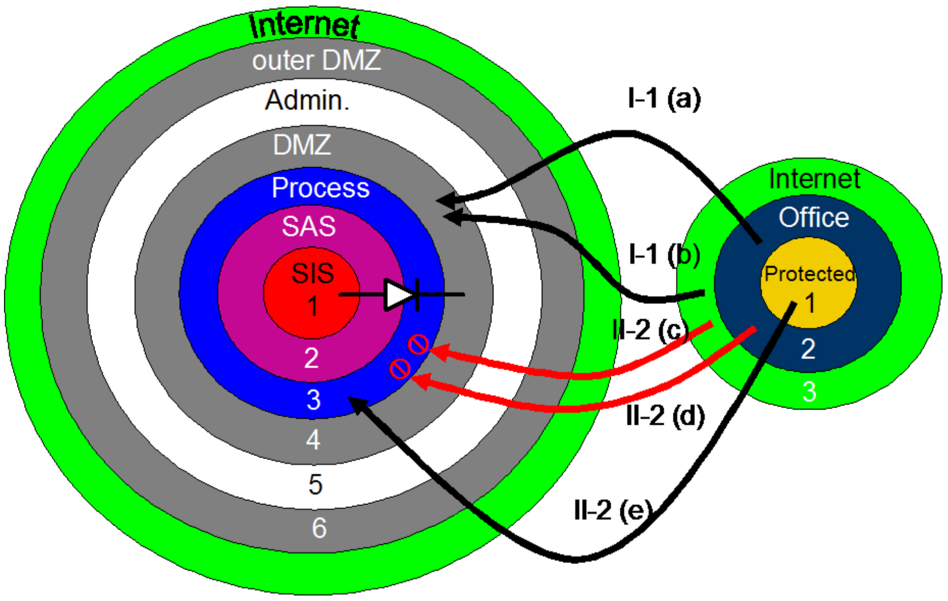

Figure 3.7: Security zones at operators and contractors

We started our work with an incident response management methodology by leaning heavily on established work such as the ISO technical report on incident management [2] (later to be superseded by an international standard [3]) and the NIST incident handling guide [20]. The ISO approach is organized around Deming's Plan-Do-Check-Act circle [15], but since we wanted to highlight the amount of effort expended on each type of activity, we decided to make our own cycle composed of three phases: Prepare, Detect & recover, and Learn (see Figure 3.8). Note that the Prepare phase in the figure is twice as big as the two others; this is meant to illustrate that most of the time, the organization will be in the prepare phase - it will only leave the prepare phase once an incident has been detected, illustrated by the exploding bomb. After the incident has been contained, and operations restored to normal, it is important to spend (at least) the same amount of effort on *learning* from the incident. This learning then feeds back into the Prepare phase, hopefully avoiding a similar incident another time. The Prepare phase will of course also be affected by external influences such as new security technology, knowledge exchange with other organizations and sectors, and changing threat picture.

5 Survivability and Malware

It has been said that the only 100% secure computer system is one that is turned off and locked in a safe[3]. Whereas this may seem a bit extreme, it does demonstrate that very

[3]paraphrased from a quote by Gene Spafford http://spaf.cerias.purdue.edu/quotes.html

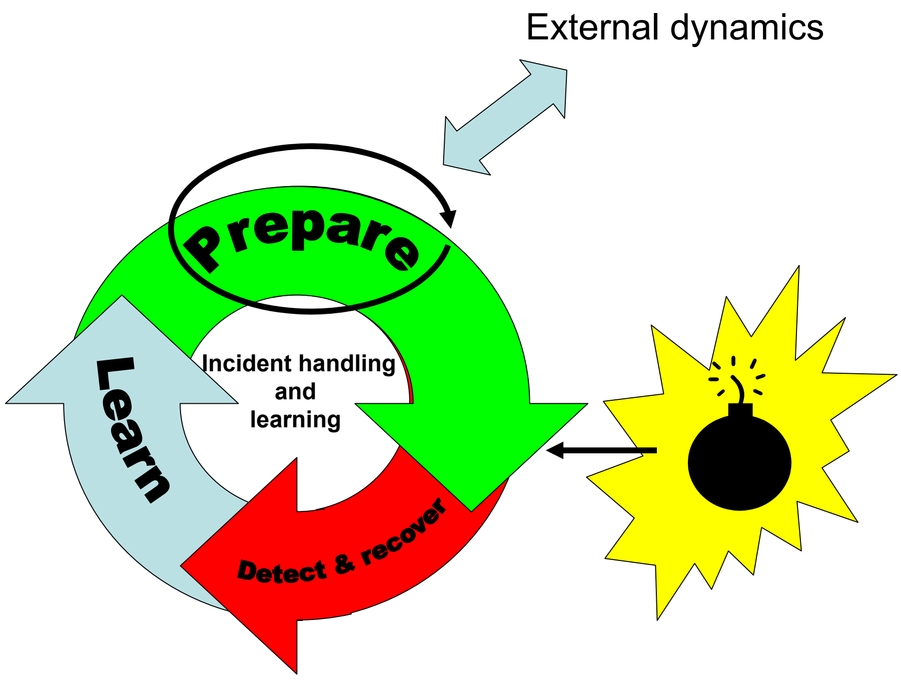

Figure 3.8: The Incident Response Management Cycle

often, real life forces us to make a trade-off between usability and security. This, in turn, implies that systems that we find useful inevitably have security flaws. Acknowledging this, we sketched [31] some ideas on how a commodity operating system could become more survivable. These ideas were then taken further at SINTEF, and a prototype of the "Increasing Survivability by dynamic deployment of Honeypots" (ISH) system produced (Paper 9 [35]).

The general idea behind ISH is to provide an internet-facing web server that is not vulnerable to application-specific or platform-specific exploits. This is done by having a very simple proxy replicating all incoming web requests, and distributing these to a number[4] of functionally equivalent, but different web servers. Each request is processed by each server, and the result compared. If all results match, this result is returned to the requesting client. If the results differ, a majority vote is used to determine what should be returned. In this case, the minority is considered suspect, and may be removed from further processing.

The premise is that well-formed requests should produce well-formed responses, as long as the web server faithfully follows the HTTP and HTML standards. Non-well-formed requests, that try to exploit implementation flaws in application or operating system, are likely to produce implementation-dependent results. In particular, this would often be true of an exploit which is successfully executed.

However, it cannot be guaranteed that all successful exploits will generate a difference in output that can be detected by the voters, and thus the ISH prototype also provided

[4]The prototype system implemented three servers.

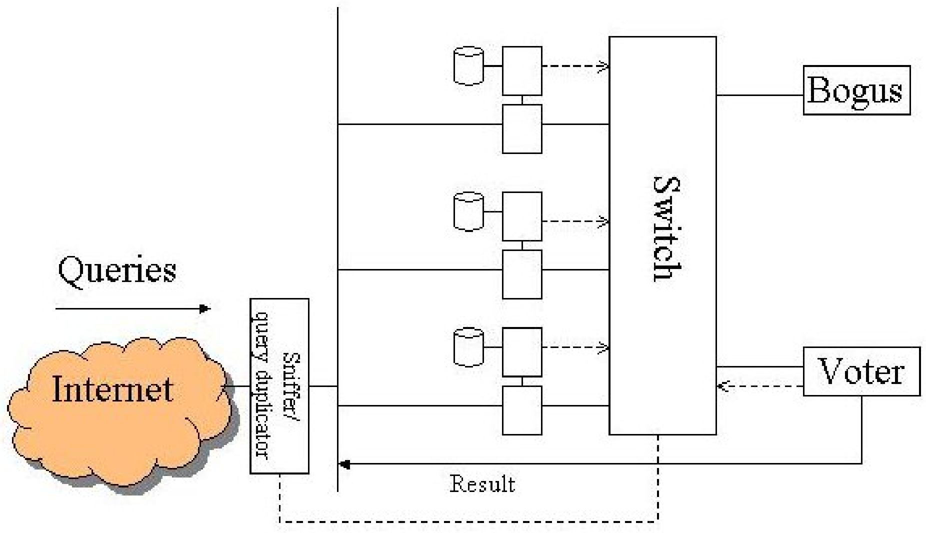

Figure 3.9: Main idea sketch of the ISH system

another level of security: Each web server did not have its own file system, but instead mounted a network file system from a special-purpose file server (one file server per web server). These file servers also performed regular malware scans on the exported volumes; these scans were in effect off-line scans, in that the server performing the scans demonstrably did not run any software present on the exported volume. This counters any stealth effects that a rootkit might have effectuated on the web server.

Now, if a compromised web server is detected, either due to a voter mismatch or a detected malware specimen, the ISH prototype facilitates an additional intelligence gathering function. Instead of shutting down a compromised server, it is instead excluded from further voting operations, and separated logically from the other servers. The existing connections that are determined to originate from an attacker are allowed to continue, but the available data on the server is quickly exchanged with bogus data which would be of no use to the attacker. The compromised server is thus converted to a Honeypot, allowing administrators to monitor the attackers, possibly learning more about their motives and future targets. Not only will this facilitate the possibility of early warning to other sites, but also slows the attackers down by making them waste their time on a system that will not provide them with any useful data.

The necessity of finding new ways to combat attackers and malware is illustrated by the final paper in this thesis (Paper 10 [33]), which reports on an experiment that was designed to collect some real-life data about the gap that exists between when a piece of malware is available in the wild, and when updated virus signatures for that malware is available.

We installed fresh copies of Microsoft Windows XP on five PCs, applied the latest service packs, and installed anti-spyware and different anti-malware products on each machine. We then embarked on two weeks of high-risk internet behavior; visiting download sites with (possibly) pirated software, participated in file sharing networks, downloaded music and videos, and more. Furthermore, like an enthusiastic but naïve internet user, we clicked “yes” or similar on all pop-ups that appeared, and installed any missing plugins or codecs that were suggested by the software we installed.

After this active period, the five machines were shut down for one month. Subsequently, the machines were subjected to off-line scans using updated anti-malware signatures. Even though the installed anti-malware had detected numerous instances during the two-week period, this new scan demonstrated that at total of 124 unique malware specimens had infected our machines without detection. It is thus clear that users exhibiting risky behavior cannot rely on current anti-malware products for protection.

Chapter 4

Discussion

In the previous sections, I have presented a number of security solutions for various areas. However, in many cases the main challenge is not developing a working security solution, but rather convincing businesses and individuals that they need it. In the following I will discuss the following recurring sentiments:

- We don't need security
- Everyone wants security, but nobody wants to pay for it
- We Cannot Afford Security

I will then dwell on the fact that security is a *process*, rather than a thing, and round off with a discussion on the challenges of empirical research in the security domain.

1 We Don't Need Security

Security and privacy are concepts that are even more related than security and safety, but they are not synonymous. The classic decomposition of security into Confidentiality, Integrity and Availability makes it tempting to conclude that privacy has nothing to do with the latter two, and thus should somehow be related to Confidentiality. The distinction between privacy and confidentiality can be subtle, since it is certainly true that privacy can be ensured through judicious application of confidentiality, but since real-life situations typically require us to exchange various types of personal information with other entities, confidentiality alone can never be sufficient.

In particular with respect to privacy, but also when it comes to security in general, individuals sometimes adopt the stance that "I have nothing to hide, so why should I worry about privacy?". In general security, this translates into "Who would want to steal my information?", with variants along the lines of "Our systems are so highly specialized that no outsiders have the required information to exploit them." These are all misconceptions rooted in insufficient knowledge, and in many of our projects in the oil and gas industry we have found ourself edging into a role as educators or awareness raisers.

The flip side of the privacy issue is that many users will claim to be concerned with privacy, but at the first decision point where they are forced to choose between functionality and privacy, they will choose functionality [37]. This is the reason why so many people use services like Facebook and LinkedIn, in spite of their sometimes questionable privacy policies, and why Privacy Enhancing Technologies (PETs) have such a hard time penetrating the market [37].

Another mental trap that organizations may fall prey to is to assume that a low number of incidents implies a satisfactory security posture. In our interaction with the oil industry, we have several times spun tales with potential scenarios that only a few years later emerged as a reality. Any industry that involves such vasts amount of money as in the energy sector must expect to be the target of future attacks.

Particularly when it comes to software security [32], security is necessary also in components that in themselves do not appear to be security-related. The classical example of this is the vulnerability in Adobe's Acrobat Reader, where a malformed PDF document could give an attacker remote access to a victim computer [58]. Nothing could be more innocuous than a program that does nothing more than display PDF files, but a PDF file received in an email or downloaded from the internet has unknown provenance, and represents an opportunity for an attacker to sneak malign code through our defenses.

2 Everyone Wants Security, but Nobody Wants to Pay for It

Security is never free. Even when not paying in purely monetary sense, users of security always need to accept that things take longer time, are less convenient, or are less exiting than they would otherwise be. A case in point is the recent uproar over the discovery of bogus GSM base stations (also know as IMSI catchers) in various strategic locations in Oslo [13], allegedly allowing tracking of and eavesdropping on top politicians and business executives. Numerous public figures were quickly demanding that "something be done", but ignored the fact that the UMTS system already has solved the biggest problems of GSM (no mutual authentication; weak encryption algorithms [43]), and switching to equipment that only uses UMTS, refusing to fall back to GSM, would directly address the threat of these fake base stations. However, that would also mean that this equipment would lose coverage as soon as the owner moves out of urban areas, which might not seem attractive to (e.g.) important businessmen. Thus, even though many people *are* willing to pay for security technology, they are often not willing to sacrifice the extra functionality or ease of use that added security would imply.

Improved usability of security technology is thus clearly an important topic, but user awareness may be even more important, at least in the short run. Everyone cannot be trained to be a security expert, but maybe it will be possible to teach everyone enough to know when they need to call one.

3 We Cannot Afford Security

In many cases, security is seen as an added cost. A security manager trying to ensure upper management support for investing in security measures needs to argue for a positive Return On Investment (ROI). This becomes difficult in the absence of security incidents – it could appear that the security investment was excessive or wasted. Strangely, this is very seldom an issue in the safety field, but can be explained by society's willingness to spend a lot of money if it can save the life of even one human being.

As security professionals, our challenge is to instead present security as an *enabler*, rather than a pure cost. There are some examples where security incidents have caused a renewed focus in information security; in 2001 the US Department of the Interior was disconnected from the internet by a court order, due to a lawsuit stemming from a security incident [63]. This caused a complete revamping of the internal security practices of this government department.

A related variant is "we cannot afford to wait for security", which often is manifested in statements such as "we first need to get the functionality up and running; then we can worry about the security". This results in "bolt-on" rather than "built-in" security, which implies that any fundamental architectural security flaws are not caught early on in the development process (if at all), causing a final product that either contains security flaws (when flaws are not caught) or is significantly more expensive (when the flaws are caught, requiring a redesign) [32].

Even in cases where there is a consensus within the development effort that security is important, external factors sometimes conspire against a good result. In a distributed development organization, programmers sometimes do not wait for the security requirements engineers to finish their work before embarking on implementation of modules, resulting in an end product with gaping security holes in spite of a committed security effort [60]. In large European projects it is also sometimes seen that the individual partners focus more on extending pre-existing tools (from previous projects), which implies that any central security requirements effort is doomed to failure, since the various partners already "know" what they are going to build, and do not "need" any additional security requirements input. In contrast, most of the projects covered by the papers in this thesis are smaller, more focused efforts where the security requirements generally where elicited by the same group that would eventually implement the solution[1]. This indicates to me that there is a need to focus more on security in agile development processes, in such a way that "ordinary" developers are responsible for eliciting security requirements for their own components [32]. We have made some small contributions in this area, but there is clearly a need for more research on this topic.

[1]With the exception of the huge SESAR programme [29], and also the possible exception of the OBAN project, where requirements and design of the Kerberos solution were performed by us, but where we provided support to the developers later implementing our design.

4 Security is a Process

In my thesis work for the Sivilingeniør degree at the Norwegian Institute of Technology, I implemented what I believe to be the first version of the BSD File Transfer Protocol application to include challenge-response authentication and encryption of file contents [10]. Although reportedly used internally at the Australian Defence Force Academy for some time, this implementation never saw wide use, something which I partly attribute to the subsequent development of SSH and SSL – with a secure transport layer, the need for communication security in each application is clearly reduced, if not eliminated entirely.

The point of this is that while you are resting on your laurels, the battlefield changes, and yesterday's security solutions may no longer be valid. It is therefore never possible to sit back secure [sic!] in the knowledge that you have achieved security; the only path to security is through continual improvement.

Whereas the different papers included in this thesis all contribute to a whole as illustrated in Figure 3.1, it is unlikely that they would ever be incorporated in a single monolithic system. However, this is less important, since each individual result gradually moves the state of the art a little further ahead, benefiting the community at large.

5 What is a Critical Information Infrastructure?

The US Department of Homeland Security [14] provides the following definition:

> Critical infrastructure are the assets, systems, and networks, whether physical or virtual, so vital [...] that their incapacitation or destruction would have a debilitating effect on security, national economic security, national public health or safety, or any combination thereof.

The European Commission [16], on the other hand, states:

> Information and Communication Technologies (ICTs) are increasingly intertwined in our daily activities. Some of these ICT systems, services, networks and infrastructures (in short, ICT infrastructures) form a vital part of European economy and society, either providing essential goods and services or constituting the underpinning platform of other critical infrastructures. They are typically regarded as critical information infrastructures (CIIs) as their disruption or destruction would have a serious impact on vital societal functions.

Critical Information Infrastructures are thus Critical Infrastructures that rely wholly or in part on Information Technology.

Offhand, it could be claimed that only two or three of the papers in this dissertation are related to critical infrastructure (i.e., oil & gas and aviation), however it has become clear to me that any hardware or software product might find itself as a component of a critical infrastructure system, whether it was designed with this in mind or not. The obvious example is the Windows-based personal computer platform, which is practically ubiquitous; e.g., Stuxnet [17, 50] targeted uranium-enrichment centrifuges in an Iranian power plant through infecting the Windows-based engineering system that served as the user interface to the process control equipment.

There are also other examples where, e.g., consumer-grade communication channels are used for critical infrastructure applications; we have documented that advanced metering infrastructures use IEEE 802.11-based mesh networking for collecting metering data, and conventional General Packet Radio System (GPRS) for backhaul communication [77, 43]. Ad hoc networking has been proposed as a solution to communication needs in crisis situation where the regular communications infrastructure has been incapacitated due to natural disasters [62, 19], and it does not require a great leap of imagination to see where the fast handover scheme described in Paper 2 and 3 [44, 66] could be integrated into a scheme for using WiFi and WiMax as fallback for (e.g.) GPRS communication in a Smart Grid scenario.

Smart Metering is an example of where critical infrastructure meets consumer electronics. Various home automation applications have been devised in concert with a smart meter, and while this potentially exposes consumers to new attacks, it also places a critical infrastructure component (the smart meter) in a potentially hostile environment where the attacker (the consumer) has full control over the physical installation. In this context, being able to detect and deflect an attack without alerting the attacker may be crucial, also in order to deploy physical countermeasures involving law enforcement personnel before the attacker has a chance to do serious damage. The applicability of the ISH-solution [35] may be even clearer in other critical infrastructure applications such as banking; in Norway 88% of consumers use the internet to perform banking operations [73], and an attack could potentially cripple the Norwegian society.

Software security is important for all software projects, and thus even more so for critical information infrastructure systems. However, also in this case it is important to note that the weak link in a critical infrastructure application often will not be those components that are purpose-built, but rather the commodity computing devices and software that support and extend them. I would therefore argue that improving the security of general computing can have an even greater impact on Critical Information Infrastructure security.

6 Empirical Research Contributions

The importance of empirical research is not something that was given undue attention during my engineering education at the Norwegian Institute of Technology (NTH) at the end of the 1980ies. NTH was deeply entrenched in the ideas later identified as Design Science [22], which might be described (a little disrespectfully and over-simplified) as: "Build something and then test if it works." Real programmers don't eat quiche [67], and a computer/software engineer would rather stick to numbers and facts than having to involve humans and "qualitative research".

Unfortunately, by omitting the human element, we often find ourselves in a position where we are building *the wrong thing*. This was a gradual process of awareness for myself, which is reflected in the fact that there little or no empirical research in the earlier articles in this thesis. However, we can note the there are empirical contributions from interviews and workshops in both SeSa [34] and IRMA [26] projects. Interestingly, the IRMA project is also an example of how mundane situations can impact academic plans: Just as IRMA was gearing up to extend the project by deploying the IRMA method in a working production environment, the target organization announced that it was merging with a major competitor. This merger soon proved to be such an extensive process that all non-essential projects (such as ours would have been) were put indefinitely on hold.

In general, it is always more *convenient* to conduct research that doesn't involve people. Any project that involves interviews, questionnaires, focus groups or any other kind of stakeholder interaction needs to take real people's schedules, availability, likes and dislikes into account. Real people are typically also less interested in your research than you are – even when they are generally positive, and when you can provide a convincing argument for "what's in it for them", you always have to play second fiddle to their "real job". This implies last-minute cancellations due to more important meetings and business travel, and potentially long delays in your carefully crafted project schedule.

In a recent project, we have experienced this yet again, and struggled to recruit stakeholders to workshops held in central locations such as Brussels and London. However, we did experience that when we invited stakeholders to local events, both during working hours and after work, many people took the time to attend. It was clear, however, that we needed to *give* them something – specifically, useful presentations that they could learn something new from. If stakeholders feel they are wasting their time, this leads to another form of stakeholder fatigue, and they will shy away the next time they see your name on an invitation.

In cryptography circles, Kerckhoffs' principle [48] has long been a maxim. In general terms, what this principle says is that a cryptographic system should not rely on the secrecy of its mechanism, but only on the secrecy of the key. Unfortunately, this principle is poorly understood by many, resulting in on the one hand instances of *security through obscurity* (where anyone with access to the inner workings of the system can trivially break it), and on the other hand knee-jerk reactions to the effect that "anything that concerns security mechanisms must be kept secret". The latter can create major obstacles for any empirical research effort in the security space.

7 Research Questions Revisited

The research questions presented in Section 3.1 map closely to the papers which have been briefly presented in Chapter 3. In the following I will summarize the contributions to each research question.

7.1 How can handovers between wireless access points be performed securely without incurring delays that affects VoIP user experience?

The fast handover scheme using Kerberos tickets (Paper 2, Paper 3 [44, 66]) solves this with a delay between 10 and 40 ms.

7.2 How can security functions be organized in a critical infrastructure network?

The security architecture for the OBAN system (Paper 1 [41]) and the good-practice approach deployed in the Secure Safety project (Paper 7 [34]) approach this from two different points of view. Furthermore, we have shown the importance of identifying the assets in a software system (Paper 4 [40]) before specifying security requirements (Paper 5 [29]), and elaborated on the difficulty of measuring the security of an existing software application (Paper 6 [25]).

7.3 How can a system survive a successful attack without alerting the attacker?

The ISH system (Paper 9 [35]) shows how dynamic deployment of honeypots can both average an attack, and contribute to early warning and mitigation in other systems.

7.4 How can a successful attack on a critical infrastructure system be mitigated by incident handling?

The IRMA method (Paper 8 [26]) documents how information security incidents can be handled in an oil and gas environment, with a particular focus on learning from information security incidents.

7.5 What is the threat posed by malware currently not detectable by signature-based anti-malware systems?

Our experiment (Paper 10 [33]) showed that aggressive and uncritical internet behavior will leave a user exposed to malware threats, even when using updated anti-malware tools. Our two-week activity caused our five test computers to be infected with a total of 124 unique malware samples undetectable at the time. This confirms that not only is current signature-based anti-malware unable to protect against a targeted attack with custom-built malware, but users who exhibit high-risk internet behavior will with high probability suffer infection also from indiscriminately spread malware.

Chapter 5

Conclusion and Further Work

Security is never a "fire and forget" concept[1]; security is a continuing process. Since the only attainable goal is "good enough" security, this also implies that security is forever a moving target: Security can only be achieved through continual improvement. In the previous pages, I have presented a number of security solutions that were considered appropriate at the time, but were never presumed to be the final solution to any of the problems they tried to solve.

I am currently working in primarily three areas: Security in Cloud Computing [69, 8, 28, 27, 82, 45, 75, 57, 7, 9, 81, 36, 83, 61, 6, 38], Security in Smart Grid [56, 77, 43, 53] and Security in Air Traffic Management systems [29, 49]. Additionally, I retain an interest in Software Security "for the rest of us" [25, 46, 47, 32, 60, 70, 78, 40, 76], and I am particularly excited about a recent opportunity to employ the BSIMM framework to survey the software security maturity in Norwegian Public Bodies.

Although one might argue that "the Cloud" is becoming more like critical infrastructure for every day, the most straightforward continuation of my work in critical infrastructure will be performed in the Smart Grid domain. Furthermore, I have recently been involved in several projects related to security of process control systems in water and wastewater distribution networks [39], an area which to the casual observer might seem less exciting than the oil industry, but which clearly is an important part of critical infrastructure, and which has its own set of interesting security problems.

[1]Although I have previously argued that it should perhaps be "forget and get fired"

Paper 1:
A Security Architecture for an Open Broadband Access Network

Paper 2:
Secure Fast Handover in an Open Broadband Access Network using Kerberos-style Tickets

Paper 3: Extending 3G/WiMAX Networks and Services through Residential Access Capacity

Paper 4: Covering Your Assets in Software Engineering

Paper 5:
Sink or SWIM

Paper 6:
Hunting for Aardvarks: Can Information Security be Measured?

Paper 7: Secure remote access to autonomous safety systems: A good practice approach

Paper 8: A framework for incident response management in the petroleum industry

Paper 9:
Survival by Deception

Paper 10: Fools Download Where Angels Fear to Tread