VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

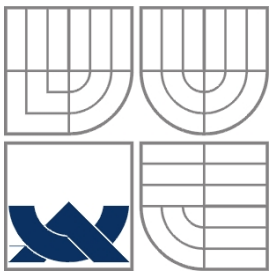
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MOBILE DEVICES ATTACKS

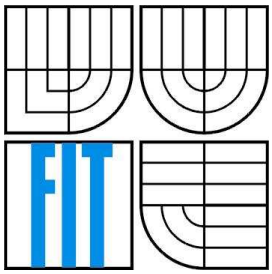
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE PETER TREBULA
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MOBILE DEVICES ATTACKS

DIPLOMOVÁ PRÁCE
MASTER‘S THESIS

AUTOR PRÁCE — PETER TREBULA
AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL OČENÁŠEK
SUPERVISOR

BRNO 2007

Abstract

This thesis studies security architecture in mobile devices and different forms of attack against them. The first part introduces the mobile devices security and security threats related to mobile devices. WLAN security threats are introduced, Bluetooth technology is described and security threats related to it. The second part introduces Nokia production testing and description of the tests which are used to proof the device stability and functionality. In the second part is also description of whole device security related to production testing and software installing.

Keywords

Security, Bluetooth, WLAN, Symbian, Attack

Abstrakt

Táto práca sa zaoberá bezpečnostnými architektúrami v mobilných zariadeniach a rôznymi formami útokov proti nim. V prvej časti je úvod do bezpečnosti mobilných zariadení a bezpečnostné riziká súvisiace s mobilnými zariadeniami. Sú tu uvedené slabé miesta vo WLAN sieťach a úvod do Bluetooth technológie aj s rizikami. V druhej časti je predstavenie produkčného testovania, ktoré sa využíva u spoločnosti Nokia a popis jednotlivých testov používaných na vyskúšanie funkčnosti zariadení. Rovnako sa v nej nachádza popis architektúry, ktorou sú mobilné zariadenia u spoločnosti Nokia zabezpečené voči rôznym formám útokov viažucim sa na inštalovanie softwaru a testovanie.

Klíčová slova

Bezpečnosť, Bluetooth, WLAN, Symbian, Útok

Mobile Devices Attacks

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Pavla Očenáška a Associate Professor, Gunver Majgaard.

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jméno Příjmení
Datum

Acknowledgement

I want to thank my supervisors, Ing. Pavel Očenášek and Associate Professor, Gunver Majgaard, for their guidance of this thesis.

Contents

1 Introduction 8
1.1 Research Motivation 8
1.2 Project Objective 9
1.3 Project Overview 9
2 Mobile devices overview 11
2.1 Devices that run with Symbian OS 11
2.1.1 General description 12
2.1.2 File types 12
2.2 Devices that run with Microsoft Windows CE 13
2.2.1 Overview 13
2.2.2 Devices 14
2.3 Summary 14
3 Types of attack 15
3.1 Viruses 16
3.2 Worms 19
3.2.1 Cabir worm 20
3.2.2 Lasco worm 22
3.2.3 Comwar worm 23
3.3 Trojan Horses 24
3.4 Backdoor 27
3.5 Logic bombs 28
3.6 Zombies 29
3.7 Conclusions 29
4 Bluetooth 30
4.1 Technology 30
4.2 Security threats 32
4.2.1 Snarf attack 33
4.2.2 BlueBug attack 33
4.2.3 Backdoor 34
4.2.4 Blueprinting 34
4.2.5 Bluesmack attack 35
5 WLAN 36

5.1 WEP insecurity 38
5.1.1 Evil twin base station 38
5.1.2 Wardriving 38
5.1.3 Long distance attacking 39
6 Attack experiment 40
7 Controlling the security threats 43
7.1 Trusted user interface 43
7.2 Security module 44
7.3 Smart cards 46
7.4 Product development phase 47
7.5 Mandatory security 51
7.6 Symbian Platform Security 53
7.7 Protection against malicious programs 56
7.8 Protection against Bluetooth attacks 56
7.9 Protection against WLAN attacks 57
7.10 Software updates 58
7.11 Operator security boundary 59
7.12 Security policies 59
8 Private part ..... **Chyba! Záložka nie je definovaná.**
9 Summary 62
Bibliography 64
Appendix 70

1 Introduction

Today's mobile devices are much more than just phones. They are versatile communication devices, which are also used to store personal data including contacts, SMS messages, emails, and calendar data. Some corporate users may carry sensitive data about their company in their mobile devices. Nowadays it is common to transfer data wirelessly for example with Bluetooth and WLAN technologies. Some advanced mobile devices can offer similar remote working capabilities as PCs do. Unfortunately, this introduces also new security threats.

During the last years, the first malicious programs emerged that were created for Symbian smart phones and Windows Mobile Pocket PCs. Mobile devices are starting to get interesting to many malware writers because of their large user base and their connectivity. During last two years many Bluetooth security holes have been discovered. Most of them disclose the personal data from vulnerable phones. WLAN connectivity in mobile devices introduces same kind of security threats as there are in PC environment.

One important difference between PCs and mobile devices from security point of view is that the mobile devices are mobile compared to stationary PCs. Mobile devices travel around the world with their users. There is also a possibility to make money with malware running in a mobile device. These facts surely interest many malware writes when they think scenarios how their malware could spread and what could they do. Also the number of mobile devices is starting to get bigger than PCs.

1.1 Research Motivation

A mobile device can be an attractive target for attacks, and security problems would seriously decrease credibility and public image of these devices. Until recently, mobile devices have been closed platforms. Users have not been able to download and install new device manufacturer or

third party software to their devices, but in open platforms, like Symbian devices, is it possible. Open platform enables new business models and opens new possibilities for platform variation, but this also endangers devices for many kind of viruses and intrusions, if the security of the system is not at the adequate level.

Hopefully, by understanding the treat and by good designed security architectures we can better secure these systems from adversaries.

1.2 Project Objective

The main purpose of the project is to research different security threats related to mobile devices. Especially threats related to malware, Bluetooth, and WLAN. This work includes studying what kind of damage they do and what could be done to prevent them. One goal is also that the reader should have a better view on security related issues after reading this project.

The first objective is to introduce mobile phone devices with open operating system Symbian and Microsoft CE.

The second objective is to describe different security threats and malware types with their examples that have been designed for mobile devices.

The last objective is to define security architecture which can be used to prevent mobile devices against illegal attacks.

1.3 Project Overview

This thesis is composed of 9 chapters. Chapter two provide basic information about mobile devices and their operating systems, focusing on mobile devices that run with Symbian or Microsoft Windows CE operating system. Chapter three introduces different malware types with examples of malware that has been designed for mobile devices. Chapter four provide information about Bluetooth and describe attacks

which are using this technology. Chapter five gives a brief background to WLAN technology and discusses WEP insecurity. Chapter six is describing Bluetooth experimet which was done to test the BlueBug attack with Blooover Java application. Chapter seven discusses options that could be done in order to control the security threats. Also the Symbian Platform Security is introduced in this chapter. Chapter nine is the conclusion chapter, which discusses the limitations, summary and future work for this research.

2 Mobile devices overview

This chapter introduces the mobile devices and their operating systems, focusing on mobile devices that run with Symbian or Microsoft Windows CE operating system. The term mobile devices used in this project include smart phones and Personal Digital Assistants. These are converging closer and closer together. Many smart phones have PDA-type functionality and likewise many PDAs have phone capabilities. The other operating systems used in mobile devices such as Embedded Linux and Palm OS are not discussed here.

2.1 Devices that run with Symbian OS

The Symbian was established as a private independent company in June 1998 by Nokia, Motorola, Psion, and Ericsson. Currently it is owned by:

- Nokia with 47.9 % of shares.
- Ericsson with 15.6 % of shares.
- Sony Ericsson with 13.1 % of shares.
- Matsushita (Panasonic) with 10.5 % of shares.
- Siemens with 8.4 % of shares.

From the beginning, the goal of Symbian was to develop an operating system and software platform for advanced and data-enabled phones. The EPOC operating system, that Psion had developed, was a solid foundation for Symbian OS. The EPOC was a modular 32-bit multi-tasking operating system for mobile devices. Nowadays there are many mobile phone manufacturers that have licensed Symbian OS and they can use it in their own products. Symbian develops new versions of the operating system, but the licensees can modify the user interface and applications for their own needs.

2.1.1 General description

The EPOC operating system was developed for small mobile devices that have all the essential office tools and use a battery for power source. This set the requirement for low power consumption and long operating time. It was developed in mid-1990s with strong object-oriented approach and used C++ programming language. The Symbian OS was developed further from EPOC to suit better for mobile phones. The emphasis has been on developing communication protocols, user interfaces and other parts of the system. The modular microkernel-based architecture allows this. [1]

Symbian develops and licenses the Symbian OS that contains the base (microkernel and device drivers), middleware (system servers such as window server), a large set of communications protocols, and a test user interface for the application engines of the operating system. Licensees develop user interfaces and application set for their own needs on top of Symbian OS. They can also license them forward to other manufacturers as Nokia has done with Series 60. [1]

For a programmer the Symbian environment offers the possibility to make applications for example with C++ or Java programming languages. Symbian has made some extensions to standard C++ to support better the mobile environment. The C++ language offers the best possibilities to make applications, as it is the native language for Symbian. [2]

2.1.2 File types

This chapter explains some of the different file types used in Symbian. It is useful to understand what these mean as they are mentioned in chapter 3 that explains different malicious programs. A Symbian application typically consists of many different files including executable, binaries and resource files for supported languages. They are all packed into a SIS-file that can be installed to a Symbian device. If a programmer wants to install his own C++ application into a Symbian device, he must first make a SIS file of that application. [1]

The Symbian OS contains four types of executable files:

- EXE-files are traditional executable files. They are usually normal console applications that don't utilize the graphical features of the operating system. Also the server programs are usually EXE-files.

- DLL-files are dynamically linkable libraries that can offer services to other kind of software components. There are two kinds of DLLs: Static DLLs and dynamic DLLs. The static DLLs offer predefined services and they are loaded into memory at application startup. Dynamic DLLs offer memory saving. While the program is running the dynamic DLL can be loaded to memory when needed and removed when it is not needed anymore.

- APP-files are Symbian applications that use the Symbian application framework. It can be also said that the APP-files are special kind of DLLs that implement the needed interface between the application and the operating system.

- EXEDLL-files are EXE-files in multithreading environment like in real Symbian devices, but in emulator that runs in Windows environment they are DLL-files.

2.2 Devices that run with Microsoft Windows CE

2.2.1 Overview

Windows CE is small footprint (200kB and up) operating system with hard real-time capabilities. It runs on x86, MIPS, ARM, and SH4 processor architectures and there is no reliance on BIOS or PC architecture hardware. Microsoft officially entered the embedded device market in November 1996 with the release of Windows CE 1.0. Windows CE has been optimized for mobile devices such as PDAs and smart phones.

Recently Microsoft has announced that it is uniting the smart phone and Pocket PC categories under the name of Windows Mobile as the technologies are merging closer and closer together. In the future these devices will be sold as Windows Mobile devices instead of separately as smart phones and Pocket PCs. Most of these devices will have phone capabilities. [43]

2.2.2 Devices

Windows Mobile-based devices include smart phones, Pocket PCs, and Portable Media Centers. However the Portable Media Centers are out of this project scope, because currently those devices don't have any kind of wireless connectivity.

A Windows Mobile-based smart phone integrates PDA-type functionality into a voice-centric handset. They are designed for one-handed operation with keypad access to both voice and data features. Pocket PC is a handheld device that enables features such as e-mail, contacts, appointments, multimedia, games, web browsing and exchanging text messages with MSN Messenger. Some Pocket PCs have also phone functionality.

2.3 Summary

Other mobile device operating systems than Symbian and Microsoft CE have been safe from malware at least for now. It seems that malware writers choose the most popular operating systems for malware platform. This way they can ensure, that the malware they have written, will the maximum amount of "customers". For example in PC environment the Linux and Mac operating systems have been quite safe from malware, while the Windows OS environment has been the main target for malware writing. Same kind of behavior can be seen in mobile device world, where the Symbian and Microsoft CE operating systems are currently most popular of the mobile device platforms and the main targets for malware writing. Currently, most of the malware that have been written for Symbian OS have been targeted for Series 60 platform. These malicious programs use features that are unique to Series 60 platform and they don't work in UIQ or Series 80 and 90 devices. Most of the malware that has been written for Microsoft Windows CE has been directed to Pocket PC devices.

3 Types of attack

This chapter introduces different malware types with examples of malware that has been designed for mobile devices. As mobile phones are advancing and because they are so popular, they are starting to get very interesting to many malware programmers. Malware is a common name for all kinds of malicious software including viruses, worms, Trojan horses, and jokes.

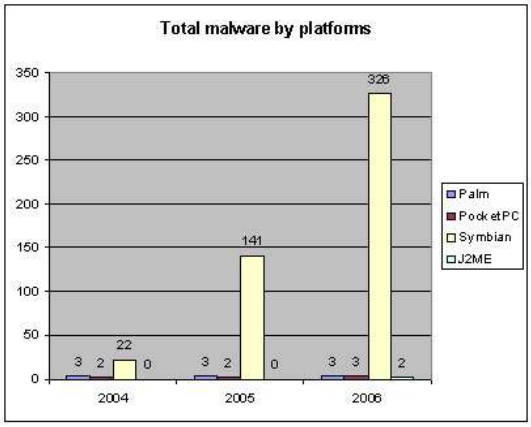


Figure 3.0.1 Malware by platforms [71]

The different types introduced here can be divided into two categories: those that replicate and those that don't replicate. Viruses, worms, and zombies have replication mechanisms.

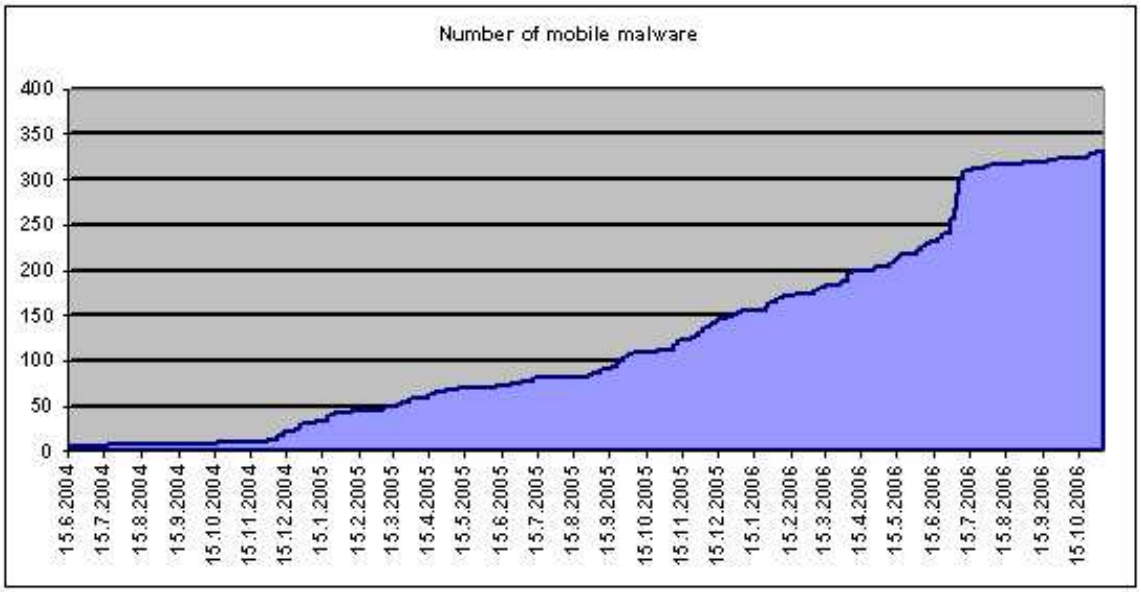


Figure 3.0.2 Mobile malware and their variants. [71]

3.1 Viruses

A virus is a program that can infect other programs by modifying them. The modified program includes a copy of the virus, which then goes to infect other programs. A typical virus takes temporary control of the device's disk operating system. When a new uninfected program comes into contact with virusinfected program, it gets a copy of the virus also. [3]

The computer viruses resemble biological viruses. Medical websites defines a biological virus as a smaller life form than bacteria (usually causer of disease) that is capable of replicating inside living cells by taking advantage of their production mechanisms. A Computer virus carries in its instructional code the recipe to make perfect copies of itself. It attached itself to other programs that it comes into contact with. When the host program is run, the virus is also secretly executed. It can perform any function what it was designed for i.e. erase files and programs. [3]

A typical virus goes through four stages during its lifetime:

Dormant phase: The virus is idle. It waits to be activated by some event for example a specific date or a presence of another file. Some viruses don't have this stage.

Propagation phase: The virus makes a copy of itself and places it to other programs or certain system areas of the disk. Now each infected program have an identical copy of the virus and they start to spread it onwards.

Triggering phase: The virus is activated to perform the function what it was designed for. Triggering phase may be caused by many system events for example the number of times that the virus has copied itself.

Execution phase: The function is performed. The function can be damaging i.e. overwriting files or harmless for example simply displaying a message on the screen.

Most viruses are written on a specific operating system or hardware platform. They are designed to exploit the security holes in systems [3]. Many of the viruses today secretly take the control over the system instead of doing some directly destructive action. This way they stay longer undetected. If the virus starts to do all kinds of visible damages to the system, the user is alarmed and he will try to remove the virus as soon as possible. There are many different types of viruses:

Parasitic virus: Attaches itself to executable files and replicates when the infected program is executed. It searches for other executable files to infect. These kinds of viruses are most common.

Memoryresident virus: Is a part of some system application that is in memory. Then it will infect all files that are executed.

Stealth virus: Explicitly designed to hide from detection of antivirus programs. These viruses can use a compression technique that makes the infected program exactly same length as the uninfected program. They can also use some more sophisticated methods for example modifying the disk I/O routines so that when somebody tries to read the suspected portions of the disk, the virus will present back the original uninfected program.

Polymorphic virus: A form of virus that mutates with every infection. The copies of the virus function the same way, but they have different bit patterns. This makes detection hard, because the virus cannot be recognized by its signature anymore.

Macro virus: Is a platform independent virus. Usually infects Microsoft Word documents. Any hardware platform and operating system that support MS Word can be infected. The macro virus can copy itself to other documents, delete files, and cause other damage to system. Macro viruses usually spread with email attachments.

Email virus: Can be activated, when the email's attachment is opened, or immediately when the email is opened. Email viruses can be categorized as viruses instead of worms, because they need human interaction to move them forward. [3]

There also exist virus creation toolkits. Using these it is easy to create new viruses and they don't require any programming experience from the user. The user can choose the wanted features from a list. The toolkit generates the virus and may add a polymorphic engine to it. Although the viruses generated with toolkits may be less sophisticated than viruses coded from scratch, they are a problem for antivirus companies because they generate so many new viruses. Today's mobile viruses can do following things:

- Spread via Bluetooth, MMS
- Send SMS messages
- Infect files
- Enable remote control of the smartphone
- Modify or replace icons or system applications
- Install "false" or non-operational fonts and applications
- Combat antivirus programs
- Install other malicious programs
- Block memory cards
- Steal data

First know virus for mobile devices is Duts. It is a parasitic file infector. Duts affects only to ARM based devices running Windows Mobile OS. It is a 1520 bytes long program written with assembly language for ARM processor.

Duts virus is a proof of concept virus. It was sent to antivirus companies by viruses author before releasing it in public. It was created to demonstrate that a virus could be written for Pocket PC environment. It does not do any other harm to device except infecting .exe files in root directory. The infected files operate normally after running the virus code.

When the infected file is executed the virus asks the user if it is allowed to spread. It displays a message box "Dear user, am I allowed to spread?" If user presses yes, the virus attempts to infect all .exe files in the device's root directory. Duts has set itself two rules in infecting: The file has to be bigger than 4096 bytes and has not been infected yet. To recognize the infected files, Duts writes a string "atar" to Windows Version field of the EXE header.

To replicate itself, Duts appends the virus body to the file and the last section is made readable and executable. The entry point of the file is set to the beginning of the virus code. Duts contains two hidden messages that can be seen with a hex editor: “This is a proof of concept code. Also, I wanted to make avers happy. The situation when Pocket PC antiviruses detect only EICAR file had to end...” Avers refer to antivirus companies. EICAR is a standard test file for antivirus programs. It is used to test antivirus programs if they operate normally. They should detect it and treat it like a real virus. EICAR comes from words European Institute of Computer Antivirus Research. Another hidden message is “This code arose from the dust of Permutation City”. It refers to a science fiction book Permutation City by Greg Egan.

Figure 3.1 Duts virus on Windows CE

Duts could be dangerous if the safety features were removed from it. The device would run out of memory soon if the virus wouldnt check the files if they have been already infected and the spreading message query would be removed. The virus would append itself over and over to files before the device would run out of memory eventually.

3.2 Worms

Worms are programs that spread from one system to another by using network connections. Once the worm is active within a system, it can perform any number of disruptive or destructive actions. To replicate itself, the worm uses some sort of network vehicle [3]. In mobile devices there are many ways to make a connection to other device and spread the worm i.e. using SMS, Bluetooth or e-mail. Advanced worms exploit the security holes in systems that have not yet been patched.

A worm has similar characteristics as a virus has. It also goes through four stages during its lifetime: A dormant phase, a propagation phase, a triggering phase, and an execution phase. During the propagation phase the worm generally performs the following functions. First it searches for all destination addresses it can find from the device where it could send the worm forward. Then it establishes a connection with the remote device. After that it copies itself to the remote device and causes the copy to be run. [3]

3.2.1 Cabir worm

Here is one example of a mobile worm named Cabir, which was designed for smart phones. Cabir worm uses Bluetooth and runs in Symbian based mobile phones. It replicates itself over Bluetooth connections and arrives to phones messaging inbox as cabire.sis file, which includes the worm. The SIS file contains worm main executable cabire.app, system recognizer flo.mdl and resource file cabire.rsc. The SIS file contains autostart settings that will automatically start the cabire.app file after the SIS file is being installed. [6] The Cabir doesn't execute automatically after it has arrived to the phone. The user must first install it to the phone and press "yes" to many questions during installation procedure.

After the user has installed the cabire.sis file, the worm activates and starts looking for other Bluetooth devices. When it finds one, it sends the SIS file to that phone and locks to it. It doesn't search for other devices even if the target moves out of range. After rebooting the phone, it will start the searching again. In Simbian phones the Bluetooth functionality is independent from GSM side. Therefore, after reboot, Cabir tries to spread even if the user doesn't enter the PIN code. [6]

The file transfer doesn’t happen automatically. The user of the target device has to first press “yes” to transfer query. Cabir finds only the phones that have BT activated and that are in discoverable mode. Cabir doesn’t find phones that are in non-discoverable (hidden) Bluetooth mode. But if the phone has been infected with Cabir worm it will try to infect other phones even if the user tries to disable Bluetooth from system settings. [6]

There has appeared many variants of Cabir-worm. Many of them just show different dialogs when the worm starts and have different filenames. The Cabir.H and Cabir.I variants are recompiled versions of the original Cabir. They have a different replication routine and they are capable of spreading faster than previous versions. While the earlier variants of Cabir were not able to search for other Bluetooth devices when the target moved out of range, the new variants start to search for other devices, and infect them as soon as the current target moves out of range. So when the original Cabir was able to infect only one phone per reboot, the new variants can infect unlimited number of phones per reboot. These new variants don’t do anything destructive or severe actions, but they prevent any normal Bluetooth connections and also they drain the phone’s battery very quickly. [12,13]

The Cabir source code has now been released on the Internet. This means that new variants will probably arise. Cabir worm is not anymore only a lab experiment. It has been already seen in the wild. Confirmed reports have come at least from Philippines, Singapore, United Arab Emirates, Mainland China, India, Finland, and Australia. [13, 19, 20]

The function that sends the worm via Bluetooth (Cabir):

```
if(WithAddress)
{
        WithAddress = 0;
        Cancel();
        TBTSockAddr btaddr(entry().iAddr);
        TBTDevAddr devAddr;
        devAddr = btaddr.BTAddr();
        TObexBluetoothProtocolInfo obexBTProtoInfo;
        obexBTProtoInfo.iTransport.Copy(_L("RFCOMM"));
```

```
        obexBTProtoInfo.iAddr.SetBTAddr(devAddr);
        obexBTProtoInfo.iAddr.SetPort(0x00000009);
        obexClient = CObexClient::NewL(obexBTProtoInfo);
        if(obexClient)
        {
                iState = 1;
                iStatus = KRequestPending;
                Cancel();
                obexClient->Connect(iStatus);
                SetActive();
        }
}
else
{
        iState = 3;
        User::After(1000000);
}
return 0;
```

3.2.2 Lasco worm

Lasco worm is developed for mobile phones that use Symbian operating system. It is the first malicious program for mobile phones that combines multiple replication mechanisms. It replicates over Bluetooth and also infects all SIS files that it finds from the phone. Lasco is based on Cabir.H source code. [27]

Lasco operates in the similar way as Cabir.H when infecting other devices via Bluetooth. It arrives to phone as velasco.sis. When the user installs it, the worm activates, and starts to search other Bluetooth devices to be infected. In addition to sending itself over Bluetooth it also inserts itself to other SIS files that it finds from the device. Velasco infects other SIS files by adding itself as velasco.sis as the last file in SIS archive. It also modifies the SIS header so that the velasco.sis will be started automatically after the original SIS file is installed. However the user has to approve first if he wants to install Velasco. [27]

The function that sends the worm via Bluetooth (Lasco):

```
if ( FoundCell )
{
        FoundCell = _NOT;
        Cancel();
        TBTSockAddr addr( entry().iAddr );
        TBTDevAddr btAddress;
        btAddress = addr.BTAddr();
        TObexBluetoothProtocolInfo obexProtocolInfo;
        obexProtocolInfo.iTransport.Copy( _L( "RFCOMM" ) );
        obexProtocolInfo.iAddr.SetBTAddr( btAddress );
        obexProtocolInfo.iAddr.SetPort( 9 );
        if ( ( iClient = CObexClient::NewL( obexProtocolInfo ) ) )
        {
                iStatus = KRequestPending;
                BluetoothStatus = _BLUETOOTH_NOT_CONNECTED;
                Cancel();
                iClient->Connect( iStatus );
                SetActive();
        }
}
else
{
        BluetoothStatus = _BLUETOOTH_CONNECTED;
}
}
```

3.2.3 Comwar worm

Comwar is the first worm to spread via MMS. Like Cabir, it can spread via Bluetooth, but MMS is the principal method used, making this worm potentially extremely dangerous. Bluetooth operates within a distance of 10 to 15 meters and other devices can be infected only if they are within this range. MMS has no boundaries and can be instantly sent even to handsets in other

countries.

Although the technology that makes it possible to send malware via MMS is the most attractive to the authors of mobile malware, so far is only known the usual transformations performed on the original worm, with baby hackers changing file names and texts in the original files without making any changes to Comwar's functionality. This is due to the fact that the source code for Comwar has not been published and the script kiddies don't know the procedure used to send infected MMS messages.

The worm looks for other sis files in the phone's memory and appends its code to these files. This provides one more propagation method in addition to the traditional MMS and Bluetooth.

It should be noted that so far Comwar has not spawned a multitude of other families. As mentioned above, the reason for this is that its source code has not been published. Just like Cabir, it is used as a carrier for other Trojan programs. Apparently, the only program using Comwar that can lay claim to having started a new family is StealWar. This is a worm that combines Cabir, Comwar and the Trojan Pbstealer. This type of combination is highly dangerous and capable of spreading widely.

Comwar also contributed to the evolution of mobile malware with a technology implemented in variant .c; this technology could be seen as rootkit technology. The worm conceals itself in the list of processes and is not visible in the standard list of applications currently running. Comwar is able to do this because its process is designated as "system". Although the process can easily be discovered using other programs for viewing running processes, this masking method is nevertheless now being used in some other malicious programs for Symbian.

3.3 Trojan Horses

A Trojan horse is a standalone program that looks genuine and harmless, but has a hidden

destructive functionality. A Trojan horse can do its malicious work immediately, after some period of time, or after when some specific conditions are met. It can i.e. alter or delete the contents of some file or destroy all data from the device. Some Trojans can disable some features of the operating system. There are also Trojans that do annoying things such as modify the icons on desktop or show some video effects that make the system practically unusable. Some Trojans leave a backdoor to the system so that a hacker can intrude to the system using it. [5] Trojan horses don't replicate on their own, but they may contain a worm or a virus that spreads them onwards.

Example of Trojan horse for mobile devices ruining on Symbian is Skulls. A skulls is a SIS file Trojan that was developed for mobile phones that use Symbian Series 60 operating system. Skulls Trojan disables all functionality from the phone except telephone functionality. You can only call or receive calls when the phone has been infected with Skulls Trojan. For example SMS and MMS messaging, camera and web browsing don't work anymore. It replaces the phone's system applications with nonfunctional versions. Skulls does not spread on it's own. The user has to install it first to mobile phone.[8]

Skulls SIS file is named "Extended theme.sis." It masquerades as a theme manager for Series 60 phones. When Skulls is installed to phone it replaces all system application. After that the icons don't refer anymore to actual applications and the phone's system applications are not able to start.

Figure 3.1 A Series 60 phone has been infected with Skulls Trojan [8]

Skulls installation file does not contain any malicious code as such. It is just a Symbian installation file that installs critical system ROM binaries to C: drive with exactly same names and locations as in the ROM drive. In Symbian operating system there is a feature that causes any file that is in C: drive to replace the file in ROM drive with identical name and location.

Another example of Trojan horse is Metal Gear Trojan. It is a malicious SIS file dropper that disables most of the wellknown third party file managers and antivirus programs and installs Cabir.G worm on the phone. Metal Gear Trojan masquerades as a Symbian version of the Metal Gear Solid game. The SIS file is named Metal gear.SIS. It spreads through pirate channels for example web sites that offer cracked versions of games and peertopeer networks.

Unlike in Skulls B and C variants that carry Cabir worms, in this the worm is activated immediately after the Metal gear.SIS is installed. Cabir worm starts to search for other nearby phones and tries to infect them with another Trojan file named SEXXXY.SIS. If the recipients accept this file and install it to phone, it disables the Symbian application menu button on their phones. The SEXXXY.SIS contains also the Cabir worm that starts to spread onwards after the installation. [10,11]

To protect itself, Metal Gear Trojan disables the following applications: Simworks AntiVirus, F-Secure Mobile AntiVirus, Application installer, Cabirfix, Decabir, FCabir, FExplorer, File manager, Smart file manager, and System Explorer. However if the antivirus program is in the real-time scanning mode it will detect the infected SIS file and prevent it to be installed. [11]

The Mosquitos game was designed for Symbian Series 60 mobile phones that have an integrated camera. The idea of the game is to kill mosquitos that fly around the screen. The game utilizes the phones camera in a way that when you move your phone, the game's background changes accordingly. It looks like there were mosquitos flying around you and they must be killed with the mobile phone before they get too close.

Mosquitos game isn't actually a Trojan, but the cracked version of it has Trojanlike qualities. The game has a functionality that sends a SMS message to a certain number each time the game is started without the user noticing it, except from the phone bill. The hidden SMS functionality was put in the game by the original manufacturer from the beginning. It was supposed to be an experimental licensing and copyprotecting technique. The cracked version can be recognized from the message that is shown when the game starts. The original version shows different message. No other differences between the original and the cracked versions have been found. [14, 15]

Current version of the game no longer has the SMS functionality, but the cracked version that is based on earlier version of the game still sends messages. These illegal versions still spread through pirate channels for example peer to peer file sharing networks.

3.4 Backdoor

A backdoor is a secret entry point into a program or system. It allows someone who knows of its existence to access the system by bypassing normal security procedures. Backdoors have been used for many years for debugging and testing purposes. A developer may wish to have some special privileges or bypass some authentication procedure. [12] This becomes a threat when an unauthorized person finds it and gets access to the system using it. Some references use the name trap door for backdoor.

A typical backdoor has a client-server structure. The server part is installed to victim's device and the hacker uses the client part. Some modern backdoors send a notification to hacker after they have activated. The notification may include the infected device's IP address and some other information. After the installation, the backdoor starts to listen commands coming from the client part. [13]

The client part is used to operate the server part. The client program has usually a GUI (Graphical

User Interface) that makes the communication with server easy. Some of the most advanced backdoors can take full control over the infected device. The hacker can for example send, receive and run files, browse the file system, get system information, retrieve passwords, get screenshots from victim's screen, change the systems date and time settings, do different annoying tricks, and shutdown or restart the system. Some backdoors can even listen for open microphones and cameras.

Finding and removing backdoors can be very difficult, because there are so many ways to create them. The only sure way to remove backdoors from the system is to restore all files from a clean backup. Good example is Brador backdoor for Windows CE. Brador is the first know backdoor for Pocket PC devices. It is written with assembly code. Brador is a twopart program. The server part is installed in to victim's Pocket PC and the hacker uses the client part. When the user receives the server part on his Pocket PC it must be executed manually. When the user does this, Brador copies the server part to Windows StartUp folder as svchost.exe. Now it is started automatically each time the device is booted. [14, 16]

Brador has not yet been seen in wild. It only runs in ARM based Pocket PC devices that have Windows Mobile 2003 (Windows CE 3.2) or newer. Brador backdoor cannot spread on its own. It does not have any selfreplication mechanisms. [14] Brador's source code is available from the Internet.

3.5 Logic bombs

A logic bomb is one of the oldest types of malicious programs, predating viruses and worms. A logic bomb is a code that is a part of legitimate program. The bomb is triggered when certain conditions are met i.e. a certain date, absence of some files or records from database. When it triggers, it can alter or delete some data, cause a system failure or do some other damage. [12] Logic bombs are in standalone programs and they don't have replication mechanisms.

3.6 Zombies

A zombie is a program that secretly takes over the victim's device. It uses the Internet connection to launch denial of service attacks. The attacks are usually targeted to some specific Web sites. The zombie programs make the tracing more difficult to the person or persons that are behind the attack. The zombie is usually planted to hundreds of devices belonging to the third parties that are unaware of the program's existence in their device. [3]

A network that consists of zombies (also known as bots) is called a botnet. The attacker, who controls this network, may use it for launching distributed denial of service (DDoS) attacks, spamming, or for collecting passwords and credit card numbers [57]. Currently these botnets are a problem only in PC environment, but if the mobile devices would be harnessed to launch DDoS attacks or spam, it would create a serious problem, because the number of bots would increase considerably.

3.7 Conclusions

The current malware examples that have emerged are not yet very severe. Most of them are proof-of-concept type programs. At least in Symbian phones those require user interaction to get installed to the phone. The user has to press "yes" to many questions during installation phase before the program is installed to phone. However at least the Cabir worm has been seen in wild in many countries. This means that the many users are unaware of mobile device security threats. The users should be informed about security threats in mobile devices and advice them not to install unknown programs to their devices.

4 Bluetooth

Bluetooth has been developed to be a cable replacement for connecting devices such as mobile phones, headsets, and portable computers. It is a short-range radio technology that is low cost and low power. The Bluetooth specification is an open and global specification that defines the complete system starting from radio all the way up to application layer.

The Bluetooth Special Interest Group (SIG) was founded in 1998. It is a group of companies working together to promote and define the Bluetooth specification. The specification has been made open rather than keeping it restricted and proprietary, because consumers are more likely to adopt a technology that works with many different manufacturers products rather than just in few. This leads to larger overall market for Bluetooth devices and this is good for all companies involved with Bluetooth SIG. [65]

Bluetooth wireless technology has got its name from a Danish Viking king Harald Blåtand. He united and controlled Denmark and Norway in tenth-century. Bluetooth name was adopted because Bluetooth wireless technology is expected to unify the telecommunications and computing industries. [16]

4.1 Technology

There are several requirements for Bluetooth technology. Being a cable replacement technology it mustn't be much more expensive than a cable or no one will buy it. Because it is designed for mobile phones, it must be able to run on batteries. Thus it must be a low power and run on low voltages. Also it has to be light and compact. It must be as reliable as the cable it replaces. So it must work well and cope with errors that happen when data is exchanged wirelessly through the air. [65]

Bluetooth devices operate at 2.4 GHz frequency band that is globally available and license-free.

This band is reserved for Industrial, Scientific and Medical (ISM) applications. These applications must obey a basic set of power, spectral, and interference specifications. There are lots of different devices using and "polluting" this frequency band for example car alarms, cordless telephones, WLAN devices, microwave ovens, and sodium vapor street lamps. For example microwave ovens use 2450 MHz frequency to heat water molecules inside food. Although Bluetooth uses frequency spectrum range from 2400 MHz to 2483.5 MHz, it isn't dangerous to humans because Bluetooth devices radiate with low output power. Also the user intercepts only a small fraction of the radio waves that a Bluetooth device emits. [16, 88]

The Bluetooth operating band is divided into 79 channels. The bandwidth is limited to 1 MHz per channel. A single message packet is transmitted on a selected channel. After that, the radio selects a new channel for transmitting the next packet. After that the radio selects again a new channel for transmitting. This technique is called FHSS (Frequency Hopping Spread Spectrum). This way Bluetooth devices use the whole available ISM band. If the transmission is compromised by interference on one channel, the transmission will be done again on a different and hopefully clear channel. [65, 19]

Although Bluetooth uses FHSS, it doesn't provide much security for data transmissions. The attacker can align with the device's hopping pattern within 2.5 to 10 seconds time.

For link encryption and authentication, Bluetooth uses a powerful SAFER+ cipher algorithm, which generates 128bit cipher keys from a 128bit plain text input. This algorithm has been released into the public domain. To use encryption, the master and slave device must share the same secret key. This secret key is never transmitted on air. This key can be built into the device (for example headsets) or it can be derived from a PIN (Personal Identification Number) entered through the user interface of the device. The PIN code used by the baseband can be up to 128 bits (16 bytes) long. It can be entered as decimal digits or alphanumeric characters where the characters are transformed into digits with UTF8 coding. [65]

The authentication and link encryption is called pairing. So that the devices could authenticate each other, they must share the same secret key that is derived from PIN. After the authentication,

the devices can create shared link keys that are used to encrypt the traffic on a link. [65]

The Generic Access Profile defines three security modes [65]:

Mode 1 is nonsecure. The devices that are in this mode do not initiate any security procedure. Authentication support is optional for devices which only support security mode 1.

Mode 2 provides service level enforced security. A channel or a service using an L2CAP connection decides if authorization, authentication, and encryption are required. The device that is in this mode will not initiate any security procedure until an L2CAP channel has been established.

Mode 3 provides link level enforced security. For a device that is in this security mode, the Link Management Protocol requires security procedures before the connection is set up. It is possible to set up devices supporting this mode so that they will connect only with prepaired devices and reject the connection request coming from any other device.

In addition to these there can be additional security modes. For maximum protection, the device can be set to a nonconnectable mode, when it isn't at use. In this mode the device will not respond to paging. This way the other devices cannot connect with it. The device can be set also to nondiscoverable mode, where it doesn't respond to inquiries. This way the other devices have to know the device's exact Bluetooth device address to connect with it. [65]

4.2 Security threats

This section describes some Bluetooth attack methods that exploit different security holes that have been found. These attacks work only on some few mobile device models. Not all devices are vulnerable. The device manufacturers have released software updates to known vulnerable models to patch the security holes. The devices that have not been patched are still vulnerable to these attacks.

4.2.1 Snarf attack

The term snarf means to take an unauthorized copy of some large document or file and use it without the author's permission [15]. It is possible to make a snarf attack on some phone models that are vulnerable to this kind of attack. The attacker can connect to target device without alerting its user and gain access to restricted portions of the stored data including the phone book, calendar, business card, properties, change log, and IMEI, which can be used in illegal phone cloning. The phone book contains the numbers associated with names and they are stored in phones memory, SIM card or in the list of missed, dialed, and received calls. The snarf attack utilizes the OBEX protocol. With this it is possible to pull known objects from the device. [21, 22, 23]

Normally it is only possible to attack devices that are in visible or discoverable mode, but there exists tools in the Internet such as RedFang, btscanner, and Bluesniff, that claim that they can find BT devices that are in "hidden" mode [22].

4.2.2 BlueBug attack

This attack is similar to snarf attack, but the BlueBug attack uses AT commands, while the snarf attack is based on OBEX protocol. On some vulnerable mobile phone models there are unpublished services on RFCOMM channels. The attacker can connect to these without pairing and issue AT commands. With these commands it is possible to do following things [21, 24]:

- Initiate phone call from victim's device to some number. The victim's phone could be turned into a bug by initiating a phone call from it to the attacker's phone. This way the attacker could listen all conversations that take place near victim's phone. Hence the name BlueBug.
- Send SMS to any number from victim's phone.
- Retrieve SMS messages from victim's phone.
- Read the phonebook entries.
- Write phonebook entries.

- Set call forward to victim's phone. The incoming calls to victim's phone are forwarded to some other number.
- Connect to Internet.

As mentioned before this attack could be used for eavesdropping when a phone call is made from victim's phone to attacker's phone. There are also financial issues involved if a phone call would be made from victim's phone to some highcost number. The phone call will be on until the user notices and terminates it. [24]

The attacker could find out the victim's phone number by sending a SMS message from victim's phone to his. Also causing financial damage to victim could be possible if SMS messages were sent to some expensive service numbers. The private SMS messages in phone's memory wouldn't be safe either, because the attacker could retrieve them from the phone.[24]

By accessing the phonebook the attacker can find out who are on the victim's phonebook. He can also find out the list of called contacts and who has called the victim. After eavesdropping the victim by calling from victim's phone to attacker's phone, he can clean his traces of bugging the phone by overwriting the last called number from victim's phonebook. [24] If the attacker uses a prepaid SIM card it would be even more difficult to trace the attacker.

4.2.3 Backdoor attack is a name for an attack that exploits a security hole on some mobile phone models. The attacker must first establish a trust relationship with the target device by pairing with it. Then he erases the entry of pairing link from victim's list of paired devices, but leaves it in victim's link key database. Now, unless the victim is observing the device at the precise moment when the connection is made, the attacker should be able to access undetected the same resources in victim's device that were available during paired session. The attacker can for example retrieve data from victim's device or access some Internet gateway such as GPRS. There are also indications that some devices that were previously immune to snarf attack, are vulnerable to it after it has been first attacked with the backdoor attack method. [22, 25]

4.2.4 Blueprinting is used for fingerprinting Bluetooth devices. It is a method that reveals the remote Bluetoothenabled device's manufacturer, model, and firmware version. This method

could be used for statistics about the manufacturers, models, and finding out how many vulnerable devices are out there. Unfortunately also some worms could use this information for distributing them selves only to devices that are vulnerable and thus remain longer undetected. [26]

The Blueprinting method creates a hash from the information it gathers from SDP profiles including the record handle and RFCOMM channel number or L2CAP psm number that the service can be accessed under. Every SDP profile entry has some unique properties that can be used for identifying the device. The device's manufacturer can be determined from Bluetooth device address (first three bytes). [26]

4.2.5 Bluesmack attack is a denial of service attack applied to Bluetooth devices. It works only to some devices. It disables the Bluetooth connectivity from vulnerable devices. This attack uses the L2CAP echo feature. If the attacker sends too big L2CAP ping (echo request) packet, for example with BlueZ utils package running on Linux OS, a buffer overflow will happen in receiving device. [21, 27, 28] There isn't much information available about this method in public, but it can be suspected that this attack crashes some broken Bluetooth implementations and the device has to be rebooted to get the Bluetooth working again. However this doesn't cause so much harm to user as the other attack methods as it doesn't reveal any personal information from the phone and it is quite easy to recover from it.

Some of the attacks described above are quite severe that disclose the user's personal data from the phone. Also they could cause financial harm to victim. It was also demonstrated that the attacker doesn't necessary have to be close to the victim. Attacks can be performed with a laptop that has a modified Bluetooth USB adapter from a long distance. Many instructions can be found from Internet on how to modify the adapter yourself. They don't require any advanced engineering skills, only some basic understanding of electronics and soldering skills. Also some readymade tools can be found from Internet to test these attacks with PC that has a Linux operating system.

Most of these attacks work only on some phone models that have old software versions. The

users that have vulnerable phones should update the phone's software in an authorized service point to patch the security holes. After this update, the user should be safe from these attacks. In product development it is important to think possible security threats and assessment their relevance, especially in Bluetooth development. Bluetooth is so versatile standard that offers many kinds of functionalities in mobile phones, including initiating a phone call, that every possible security threat scenario should be considered.

5 WLAN

Nowadays the new mobile devices are starting to have wireless LAN functionality implemented into them. This brings also new security threats and it is needed to prepare for them in advance. The WLAN issues that are discussed here are from PC environment, but the same issues apply mostly also to mobile devices.

There exist many different wireless LAN specifications and standards. The IEEE 802.11 was the first IEEE standard for wireless LANs that was introduced in 1997. This standard specifies a 2.4 GHz operating frequency and 1 Mbps and 2 Mbps data rates. Two forms of spread spectrum modulations are specified: frequency hopping FHSS and direct sequence DSSS.

The supplements to 802.11 standard were released in 1999 and they were called 802.11a and 802.11b. The 802.11b is a data rate extension to the initial 802.11 DSSS and it provides a data rate of 11 Mbps in 2.4 GHz frequency band. The 802.11a standard provides even faster 54 Mbps

data rate at maximum and a new OFDM modulation in roomier 5 GHz frequency band.

HiperLAN is a European specification that was ratified in 1996 by ETSI. HiperLAN/1 operates in 5 GHz frequency band with a maximum data rate of 24 Mbps. HiperLAN/1 also provides quality of service support and a shared access to the wireless LAN among end users via a connectionless protocol. HiperLAN/2 operates also at 5 GHz band and supports data rates up to 54 Mbps. It uses connectionoriented protocol for sharing access among end user devices and the QoS is supported.

At the time when the WLAN was designed, the designers didn't expect that the wireless LANs would be very popular and they didn't pay enough attention to security issues. The Wireless Equivalent Privacy was standardized as encryption method. The WEP uses two types of protection: a secret key and encryption. The RC4 encryption algorithm is used to cipher the data that is sent over the air. The RC4 is a very fast and simple algorithm that scrambles each byte of data that is sent in a packet. There are two types of secret keys: 64bit and 128bit. The secret key in 64bit version is made of 5character password (40 bits) that is shared between the access point and WLAN users, and a pseudorandom 3character initialization vector (24bit) that is appended to that preshared password. In the 128bit version the preshared password's length is 13 characters (104 bits) and the initialization vector is 24bit. Each sent data packet has a unique secret key, because the initialization vector changes with every packet. [88]

Because of the weaknesses that were later found in WEP a new encryption method was developed. The new technique's name is Wireless Protected Access.

The base station can have a list of known MAC addresses that are allowed to access the network. However this isn't a very good security mechanism, because the MAC address can be changed easily with software and by monitoring the base stations traffic it can be seen what MAC addresses are used in that network.[88]

Another way to increase security is not to broadcast the network segments name (SSID). This way the attacker has to know that name to make the connection to it. If the SSID broadcasting is on, anyone can try to access that network. [88]

5.1 WEP insecurity

A major weakness in WEP is that the initialization vector is only 24bits long and it is sent as sent as plaintext with the encrypted packet. The hacker can sniff the packets in network and, sooner or later, the packets with the same initialization vector will appear and the secret key can be determined. There exist readymade tools in the Internet (for example AirSnort) to sniff the packets in wireless LANs and crack the secret key if enough data is captured. [88]

Roughly 7GB of data must be captured, on average, to crack the password. For this reason the home users are quite safe from this attack. It may take nearly two weeks before this much of data is captured from a home user network. However in a corporate network this amount of data can be captured in couple of hours so the threat is real. A more detailed description about cracking the WEP can be found from Seth Fogie's and Cyrus Peikari's paper. [88]

5.1.1 Evil twin base station

There is a new increasing threat in wireless networks. The hackers can steal passwords by placing their own WLAN base station near a commercial base station that gets covered by the hackers WLAN network. This "evil twin" base station has the hackers' own server program running in it and it has stored the most common web pages that ask for passwords for example online banks and email services. The server program stores all passwords that are entered through it. [33]

The hackers only need a laptop and a base station that both can be fitted into a backpack. They can use this form of phishing for example in airports and coffee shops. They can offer web services for example if the credit card number or some other personal information is first entered to it. The most common web pages that ask for passwords are stored in server's cache. If the users want surf to other pages then a "page cannot be displayed" page can be shown. The firewalls cannot give protection against this form of phishing where the user logs in to this kind of hoax service. [33]

5.1.2 Wardriving

In the modem age the hackers were trying to penetrate into companies by calling systematically to all phone numbers in district. They were trying to find companies' remote access and terminal server numbers. It could take hundreds of attempts before those numbers were found, but at that time the local calls were free in U.S. so it didn't cost anything to hackers. This was called wardialing. [88]

Now in the wireless networks age a similar "probing" method has gotten a name wardriving. This is a method where the hacker drives around the town and tries to find wireless networks that could be penetrable. He might also have a GPS locator to store the locations where these vulnerable networks were found. [88]

5.1.3 Long distance attacking

Whichever attack method is used the attacker doesn't necessary have to be close to the target. If the attacker wants to penetrate into corporate network it may not be possible to just drive to company's parking lot and start attacking the network with a WLAN device. The corporate security guards might soon come to ask what he is doing. Instead he can attack from a long distance using a directional antenna. The attack can be performed from so long distance that the eye can't see. Some antennas are claimed to be capable of connecting to a WLAN network from over 15 km away.

In the Internet there exits many do-it-yourself guides on how to make your own directional WLAN antenna. These can be easily found with Google by typing "WLAN antenna" in search query. One of these pages that include many different instructions can be found from here [34].

The WEP encryption is known to be a weak. However, a user that doesn't have a highly trafficked WLAN network should use the WEP encryption if other stronger encryption methods, such as WPA, aren't available. The attacker that is searching for vulnerable wireless LANs will probably move to another network if he finds out that the network traffic is encrypted. The users need to be careful when logging in to unknown WLAN access points for example in airports and

coffee shops. The hackers may have established their own access point to collect passwords and credit card numbers. Caution must be taken always when this kind of information is asked.

6 Attack experiment

This experiment was done to test the BlueBug attack (see chapter 4.1.2) with Blooover Java application that can be downloaded from the Internet. Blooover is meant to be a proof-of-concept tool that runs on J2ME-enabled mobile phones. With this tool it is possible to test if some phone is vulnerable to BlueBug attack. From a vulnerable phone it is possible to read the phonebook entries, SMS messages, add a phonebook entry, set a call forward to the phone, and initiate a voice call. So that this tool wouldn't cause too much financial harm to victims, the voice call and call forward numbers cannot be edited and they are free-of-charge.

The Blooover application was installed to a mobile phone that is named here Phone A. The mobile phone, which was going to be attacked, was Phone B with an old software version. The mobile phone manufacturers have recently released software updates to phones that are vulnerable to this attack, including this Phone B. After the software update, this attack doesn't succeed anymore. First, the attack features were set from the application's settings (figure 6.1). It was set to read the first 100 entries from the phonebook and 10 SMS messages, add one phonebook entry "Honey" with a number to phonebook, and set a call. After the settings were finished, Blooover was ready to start searching for Bluetooth devices in range.

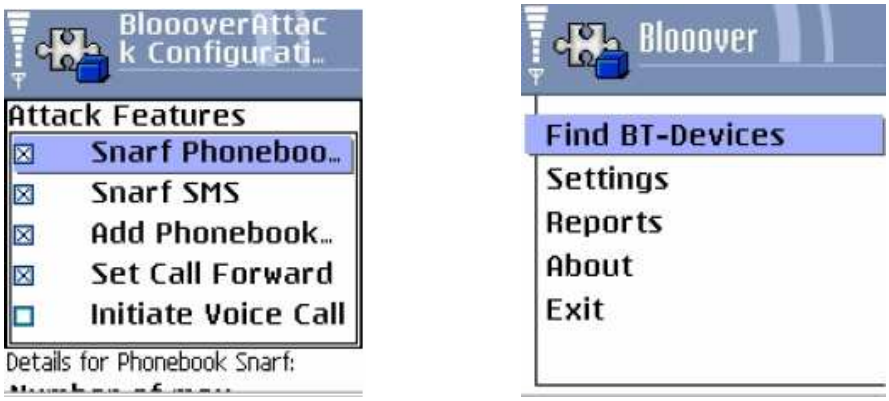


Figure 6.1 Blooover application settings (left picture) and start menu.

Blooover scanned for all Bluetooth devices it could find (figure 7.2). Blooover finds only devices that are in discoverable mode. After the scanning was complete, the target for attack was selected from a list (figure 6.2).

Figure 6.2 Blooover scans for nearby Bluetooth devices (left picture) and the list of targets after scanning.

When the target was selected, the user is asked if the application is allowed to use the Bluetooth connection. This is a Java security feature called *sandbox*. If the Java application wants to use for example Bluetooth connections, the user is queried if it is allowed to do this. This way an application cannot do any malicious activity without detection. After pressing “yes” to query, the attack begins (figure 6.3). Now, in the target Phone B, there comes a notification “Connected Phone A” on the display for a moment. Also a soft beep sound is played. If the phone would be in the pocket of the user, it would be very hard to notice that anything suspicious is going on. During the attack there is a headset icon on the display of the target phone.

Figure 6.3 Attacking the target.

After the attack is completed, the list of results is shown (figure 6.4) and the connection is terminated. On target phone there comes a notification ”Disconnected Phone A” on the screen for a moment and a loud beep sound is played. This beep is so loud that the user would probably hear it even if the phone would be in his pocket. But by the time when he reaches for his phone, the notification is disappeared from the display.

Figure 6.4 Results of the attack.

The attack was successful. The phonebook entries were read. The two SMS messages that were in target phone were read. When browsing the contacts list in Phone B after the attack, there was found the added entry “Honey”. However, the call forward setting didn’t work for some reason. When the Phone B was called after the attack, it ringed and the call wasn’t forwarded to other number.

7 Controlling the security threats

This chapter discusses some options that could be done in order to control the security threats. Also the Symbian Platform Security is introduced in this chapter.

7.1 Trusted user interface

Mobile device users need to ensure that they interact with legitimate applications and that their keyboard input cannot be read by malicious program. Trojan horses are programs that pretend to be useful programs while they in reality perform some unwanted function. It is an existing threat in mobile devices that there are programs that try to mimic or eavesdrop existing applications, services, or operating system dialogs, and steal confidential information or bluff user to commit some harmful action. There is a need for mechanism that user can distinguish fake applications and dialogs from genuine ones and trust to user interaction. Trusted user interface solutions try to guarantee trusted path between a user and a trusted computing base.

A trusted path is a security mechanism, which provides trusted input and output channels to ensure that the user interacts directly with trusted software. It can be only activated by either the user or the trusted software, and it cannot be imitated by other software. Without a trusted path, malicious software could imitate trusted software to the user or imitate the user to trusted software. For example in many open platforms, it would be rather easy to imitate any view to the user and trick the user to reveal the password of the digital signing key.

A trusted path is not enough to implement a trusted user interface. The user should be able to separate the trusted user interface from untrusted ones. Therefore, the trusted user interface should have a mechanism to indicate the trustworthiness of the user interface for user. Commonly, there are three options to implement trusted UI in mobile device: trusted screen area, hardware based indicator, and window personalization. Trusted screen area can be dedicated

from the graphical user interface to indicate trustworthiness of the user interaction and it is controlled by mandatory security mechanism. The application screen must not be able to override trusted section. In addition, the application region of the screen should be controlled by mandatory security mechanism, when establishing trusted user interaction with user. Simple hardware based trusted mode indicator could for example be LED, that is controlled by a mandatory security mechanism. Window personalization may be used when hardware based indicator or trusted screen area are not feasible due to e.g. economical reasons. For example, mobile device manufacturers do not necessarily want to reduce the size of screen or add one extra LED to the device because of trusted UI. In that case, window personalization may provide alternative solution. Window personalization must be initialized before first usage. The user can personalize trusted UI by adding some personal indicator, e.g. image or text string, which would be then used to indicate trusted mode. Although personal indicator would be stored to protected area, this solution is however vulnerable to screen capture attacks.

In mobile devices, many notes are displayed to a user using popup windows, and consequently the user does not necessarily understand the context of the launched window, which can e.g. be a password query. This may lead the user to make security relevant decisions based on wrong assumptions. For example, if the user is using two applications at the same time, the popup window could be invoked by both applications, but the user would not be able to recognize which application actually launched it. Mandatory security mechanism with trusted UI can be used to map windows in graphical user interface to domains that are understandable for user. For example, every window may contain application icon in trusted screen area to help the user to recognize the application that launched the window.

7.2 Security module

Security modules are defined to be tamperresistant devices that can perform data processing and store confidential information. If a mobile device cannot fulfill all the security requirements needed to be a trustworthy device, the most critical operations can be performed in a separate

security module, like smart card. Although a mobile device could potentially be secure enough to perform digital signing, the commercial mobile device might be so complex, that formally evaluating the security level of the device might be unreasonable task. Any certification for tamper resistance is a time consuming and expensive process. Smaller security modules enable an easier and lesscomplex verification process and they can be reused within multiples devices later on. Additionally, it may enable easier adoption to local law requirements, which may be essential for digital signing and other encryption functionalities in many countries. When using separate security module, mobile device just need to address that it can provide trusted UI and trusted path between user and security module.

Movable security module can be used to personalize mobile device for individual users. Whenever a user is changing device to another, security module provides easy and secure way to move user's personal credentials, like private keys, certificates, access point settings, tickets etc. to the new device.

Security module alone does not solve all the security problems. It requires secure application environment to communicate with. If the security module is interacting with a vulnerable mobile device, tamper resistant security module can only protect against direct reading of keys. It is assumed here that security module can be hardware or software based because same security rules apply to both.

Security module needs to identify with whom it is communicating. Hence, the invocation of the module must be protected. Without mandatory security, caller application cannot be sure that invocation is not modified, bypassed, or directed to some malicious entity that imitates the security module for the caller. Mandatory security mechanism can be used to ensure that it is not possible to bypass or modify calls to the security module. Protected path mechanism is needed to guarantee that malicious software cannot impersonate the invoking of applications or vice versa.

Other vulnerability is unauthorized usage of the cryptographic operations. If the underlying operating system cannot identify callers reliably, a security module can only ask authorization from user, but token cannot ensure for the user what application is trying to use cryptographic

service. In addition, after authorization it cannot be guaranteed that only authorized applications are using services enabled by authorization. In this case, malicious applications or intruders can misuse the security module on behalf of authorized user and application. Furthermore, without direct physical user interface for querying authorization from user, malicious application can spoof the token to the user and obtain authentication information, which can be then used to activate sensitive cryptographic service without user's knowledge. Although direct physical interface would exist in the security module, it would be inconvenient to get user authorization for every cryptographic operation, and anyhow token could ensure the identity of the caller application.

A trusted UI could be used for authorization instead of direct physical user interface. The protected path mechanism enforces caller applications to identify them to the security module and it may provide some other mechanisms to control the usage of the security module. Alternatively, mandatory security mechanism may provide such controls. For example, mandatory security mechanism may be used to isolate the usage of cryptographic services only for process that activated the token. Naturally, mandatory security mechanism may prevent malicious software to use cryptographic services of the token.

Additionally software based security module requires operating system including memory protection and file protection from the operating system, because otherwise security module cannot operate confidentially and ensure integrity of the cryptographic services. Mandatory security mechanism is also needed to protect inputs from external services to software security module.

7.3 Smart cards

Smart cards are most commonly used security modules in mobile devices. Smart cards are very small and portable devices of relatively low capacity and some tamperresistance. For example Subscriber Identity Module (SIM) and JavaCards are smart cards. Smart cards can be moved from one device to another relatively easily, thus transporting e.g. user's private keys and tickets,

and because of this mobile network operators are willing to use smart cards as a security module in mobile devices.

Smart cards are standardized according to ISO 7816 , which defines their packaging and physical, electrical and logical interfaces to the outside world. The simplest type of smart cards consists of memory only, but ones that are more complex include an onboard processor, too. Due to the cost, power supply, and chip area limitations, smart cards have a limited amount of memory, a low performance processor, and a limited set of support circuitry.

A smart card with a processor can implement active measures like counting unsuccessful attempts and block the operation if a limit is reached. This is a major benefit compared to encrypted file in mobile device's file system, which has not reasonable way to limit the count of decrypting attempts. Critical data, like private keys can be stored in smart card, which also perform cryptographic operations without transporting keys outside the card. In addition, smart cards can be used to store data that should be kept intact, like public keys and root certificates. Smart card enables easy way for operators and other card vendors to deliver small amount initialization and personalization data and user specific data to mobile equipment, e.g. root certificates, personal certificates, private keys, and access point settings.

For example, operators may want to deliver their root CA certificates in smart card, and they may configure it undeletable to confirm that the user has always possibility to establish security connection to operator's services. If there are many certificates, there may be a need to store them in the phone, because certificates are often quite big compared to the capacity of the smart card. If the user uses one smart card in many mobile phones, it may be useful to store certificates smart card, so that he or she can always have the same important certificates in use.

7.4 Product development phase

In product development of mobile devices, the security risks should be carefully analyzed already in the beginning of the project. As this project has pointed out, there are many existing and many

potential security threats related to mobile devices. Therefore the security of mobile devices should be analyzed carefully. If it is ignored or implemented badly, it may quickly produce bad publicity to the company and the lost of trust of customers. It is easy to lose the trust of customers and hard to get it back.

The principles of good security can be easily understood, but it is challenging to implement them fully in the real world. If the security issues are taken into account when designing the device's operating system, it creates a good basis for system security.

Designing security is a challenging task. The security measures should be hidden as much as possible from the users. If the users are required to configure complex settings and answer to multiple queries, they will easily ignore the security measures. It doesn't make sense to design fancy security features if they are too complex and frustrating for average user to use.

The installation warning dialogs like "Installation security warning. Unable to verify supplier. Continue anyway? Yes – No" don't provide much security against malware. If most of the installable files show this dialog, most users will press "Yes" to that dialog without stopping to think if this is potentially malware.

Historically, security specialists have been impassive to usability, and until lately there have not been many publications or research projects concerning security and user interaction. However, the correct use of software is just as important as the correctness of the software itself. Garfinkel and Spafford have given good definition: "A computer is secure if you can depend on it and its software behaves as you expect."

Mobile device's security solutions and corresponding UIs have been more or less copy pasted from the desktop environment, and mobile device's limited capabilities have not been considered adequate. Most end users are not familiar with security technologies and the associated terminology and vocabulary. It is common understanding that security applications are too complicated for end user, and because of their abstract nature, security aspects are difficult to understand. Many designers habitually assume that improving security necessarily degrades

usability and vice versa. Designers have not bothered enough to rethink how to take the security feature's UI to the level of end user. Commonly used security fix is to show confirmation query or warning dialog for user. Quite often they are just annoying users, and they do not even bother to read it. In many cases, these dialogs and queries could be prevented by using smarter security design. When designing a security system, important objective is to design intuitive user interaction that does not require excessive effort from user. The feature is useless for the user if one is not able to use it right.

There seem to be relatively few development efforts in computer security that have seriously emphasized user interaction issues. Until now the most fundamental publication is KaPing Yee's article "User Interaction Design for Secure Systems" [9] which presents design principles of secure user interaction.

Principle of Path of Least Resistance describes the physical or metaphorical pathway which provides the least resistance to forward motion by a given object or entity, among a set of alternative paths. The concept is often used to describe why an object or entity takes a given path. The most natural way to do any task should also be the most secure way. Users are primarily concerned with accomplishing some useful task. They do not want to spend their time thinking about security. Hence, user's motivation and the security goals should be aligned with each other. The ultimate path of least resistance is for the user to do nothing. Therefore, the default settings for any software should be secure. In addition, user interface should not lead the user to the situation to work against security unintentionally or intentionally.

In *Principle of Appropriate Boundaries* the user interface must distinguish between objects and actions along boundaries that matter to the user. If the distinctions are too detailed, there is a risk that users will overlap or leave out details. On the other hand, if boundaries are too general, users may be endangered to authorize more than they intend. Any boundary that could have meaningful security implications to the user should be visible, and those that do not should not be visible. The right distinctions can be discovered by answering following questions: Would the user ever want to manipulate this authority independently of another? To grant an authority to this actor but not another? To permit access to this resource but not another?

Principle of Explicit Authorization is the most basic requirement for controlling authority in any system. If each user's authority is explicitly granted, it is likely that actors will operate with the least authorities necessary. It may look like that the principle of explicit authorization is in conflict with the principle of the path of least resistance. It might be easily thought that the principle of explicit authorization means that we have to constantly intercept the user with annoying security prompts to confirm every action. Actually, this is not true. For the most of the time, extra confirmation is not needed, if we just utilize all the information the user has already provided. For example, if the user enters URL to web browser, it is already clear that the user wants to make a connection to a network. Further confirmation is unnecessary.

Principle of Visibility. Human resources are limited. A user may forget some of the granted authorizations, or a user can face unfamiliar state, where the user feels itself uncertain. Therefore, the user should be always able to check the actor ability state of the system at any time. It should be possible to review what each actor can do; otherwise, the user may make security relevant decisions based on wrong assumptions. In addition, any authorized application could use the authority unobserved, if user has forgotten the granting action. For example, if verification of user stays valid for a limited time, the user must be aware of this. Anyhow, there is a risk that other applications may try utilize the granted authority during the validity period. For example, other applications may try to delete all the essential data protected with password if the verification of the password stays valid for a while.

Principle of Revocability. People make mistakes and wrong decisions. Therefore, whenever possible, the user interface should always allow the user to revoke granted authorities. For example, a firewall should the contain possibility to revoke network access from an application. In addition, in order to be able to control the growth of the actor ability state, the user must be able to revoke authorizations.

Principle of identifiability is needed to protect user against being fooled. Identification follows two concepts: continuity and discriminability. The user interface should be designed show that the same things appear the same (continuity) and different things appear different (discrimination). One basic requirement related to this is that trusted UI components should be

clearly distinguishable from other UI components.

Principle of Expressiveness means that a system must be able to express the policy that is wanted. This principle applies both to the user interface's capability to express the security policy to the user and to the capability for the user to express security policy to the system they want. If the system does not enable configuration of some security policy how it is intended, this principle does not realize, i.e., the user or security administrator is not able to tell to the system what they want.

Principle of Clarity address that effects of any security relevant action must be clearly apparent to the user before the action is taken. In other words, the user knows what consequences of security relevant decisions are. For example, if user removes some certificate or associated truster application in certificate management, information of the consequences should be available for the user. It might be that after the removal the user is not able to install additional Java midlets, or it is impossible to establish a secure connection to some service if client application accepts only server certificates verified with root certificate.

When the security risks have been analyzed and the security measures have been designed, it would be good to have some 3rd party security expert to review them. Sometimes when working on some problem for a long time, it may blind the designer(s) from some obvious things. The 3rd party may bring fresh point of views and opinions. If the 3rd parties are not wanted to be used for some reason, it would be good to have as many internal workers as possible to review the security issues for possible security holes and vulnerabilities.

7.5 Mandatory security

Mobile devices can be used to authorize many kinds of transactions, such as banking, payments, ticketing, and secure access based operations. When mobile devices are transforming from wireless telephones to trustworthy mobile user devices, certain security requirements need to be fulfilled in order that device can be thought secure enough environment.

Security policy control can by divided to discretionary security and mandatory security. Mandatory security policy is tightly controlled by a system security policies are complied in the whole system. For example, nonrepundation of origin service requires mandatory security policy with private keys, because even user should not have possibility to access them. Discretionary security can be controlled by user.

An operating system's mandatory security policy may be divided into several kinds of policies, such as an access control policy, an authentication usage policy, and a cryptographic usage policy. A mandatory access control policy, it specifies how subjects can access to objects in the operating system's environment. Subjects are any active entities of the operating system, such as users and processes. Objects are passive entities of the system, like files and programs that actions are performed upon. A mandatory authentication usage policy specifies what mechanism must be used to authenticate a party to the domain. A mandatory cryptographic usage policy defines what cryptographic mechanisms must be used to protect data. For example, it can specify that every digital signature must be explicitly authorized by the user.

If mandatory security mechanisms of the operating system are unbypassable and tamperproof, they can be used to support security related functionality in applications. Trusted applications are programs that require special security privileges to perform some functions in mandatory security system. They are called trusted, because they are trusted to perform some action that is somehow constrained because of security policy. They are also trusted not to misuse the granted rights. Applications and other subjects should have the minimal set of rights required for their functions in order minimize the damage caused by misuse of the rights. This common security concept is called the principle of least privilege.

Mandatory security policy may be enforced through discretionary security mechanisms, but it would not protect against malicious or careless users. Carelessness may lead to a violation of mandatory security, and security could be still vulnerable against flawed and malicious software in any case. Although user could have been carefully defined the discretionary policy to implement mandatory policy, an application might still change the discretionary policy without user's awareness. In contrast, mandatory security mechanisms can be used to ensure that security policy

is followed and it can protect the user and other entities of the system against bogus execution of untrustworthy application. In addition, mandatory security policy cannot be changed without the system security administrator.

7.6 Symbian Platform Security

The new Symbian OS version 9.5 is good example of a secure operating system where the security aspects have been taken into account in designing phase. This Symbian OS version introduces the Platform Security concept. The Platform Security ensures the integrity of the phones by controlling the capabilities of applications that are installed to the device. The applications can't access any more to all resources on the device, if that application isn't trusted enough. [77]

The Platform Security architecture (figure 7.2) can be divided into three layers. In the core is the Trusted Computing Base, which is responsible of maintaining the integrity of the device. It has unrestricted access to all the devices resources. For example the kernel, file server, and the software installer are a part of the TCB. All the components that have TCB capabilities must be very carefully reviewed that any malicious code doesn't get into here. Otherwise the malicious code would have unrestricted access to any system's resources. Surrounding the TCB is the Trusted Computing Environment. It consists of important system servers. These system servers control the access to shared resources and thus protect these resources from misuse. Surrounding the TCE are for example the communication protocols and the 3rd party applications. [77]

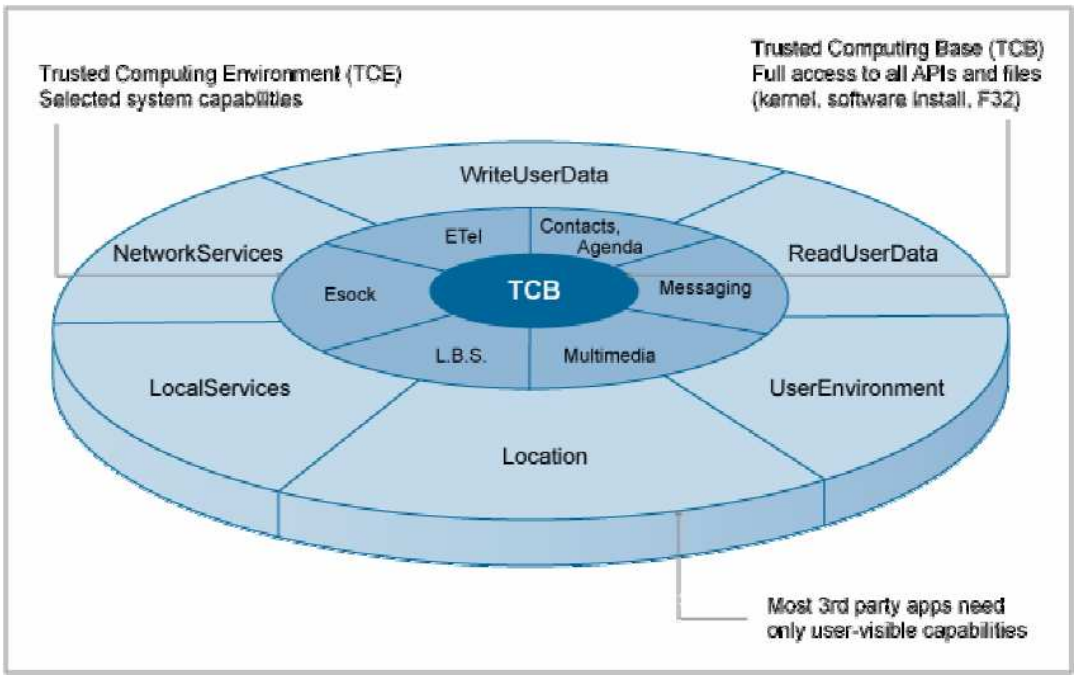


Figure 7.2 Symbian Platform Security architecture [77]

As mentioned before, the Platform Security controls the capabilities. It can be said that a capability is a token that corresponds to permission to access to a sensitive system resource. The security architecture provides different capabilities for example access to phone stack or to complete file system. A client program that requests access to a system resource must have the same capability as the server that provides that service. [77]

The Platform Security has data caging functionality. This allows applications to have a private data storage that is not accessible by other applications. The system directories are also inside the data cage and they are not accessible anymore to applications unless the applications have very high capabilities. The security architecture enforces the data caging. [77, 78]

A cryptography module is also included in Platform Security. This includes many different cryptography algorithms and support for different ciphers, which enable the encryption and decryption of data. Also a cryptography token framework is included that the licensees can integrate support for removable hardware devices such as WIM modules. The certificate module

is used to authenticate other entities for example the 3rd party developers and web services to the user of the phone. Also the user can be authenticated to the phone with this module. [77]

Digital Rights Management is supported in Platform Security. With this it can be controlled that only authorized content consumers have access to protected content. In other words it can be said that only users that have paid for the content have access to it. The following content file types are supported: GIF, BMP, JPEG, PNG, JAR, SIS, MIDI, MMF, AMR, and WAV. [77]

The software installation process has been developed to be faster and more secure than before. When installing C++ executables and Java MIDlets, they are authenticated with digital signatures that provide a measure of confidence that they are from known reputable vendor. The install packages can also be compressed to reduce disk space and download times. There is also a feature of different varieties of phones that the installation package creator can ensure that the package is installed to an appropriate phone. [77]

The capabilities of an application are granted and monitored trough the Symbian Signed certification. If the developer wishes that his application could access the sensitive system resources, he must get a certificate first. To get the Symbian Signed certificate the developer must be individually identified and the application must pass certain tests. After this the developer can obtain a digital signature for their applications. If any problems occur later on, the developer of that application can be traced back. [78, 79] Writing malware for Symbian OS version 9 becomes very hard. It shouldn't be possible anymore that the malware could access for example the critical file system areas or telephone functionality without the certificate that grants these capabilities.

There is also one drawback in this new Symbian OS version. The binary compatibility has been broken in this version. The existing applications won't work in this new OS version unless it is recompiled for the Symbian OS version 9. Also if the software that has been developed for Symbian OS v9 is to be used in previous OS versions, it must be recompiled with an older SDK. [78]

7.7 Protection against malicious programs

The Symbian Platform Security helps to give protection against malware. However, an anti-virus program should be used to give extra protection against malware. Also all the other mobile devices that don't have a secure operating system should have an antivirus program installed into the devices. The anti-virus programs should have some kind of update mechanism, so when new malware is discovered, the anti-virus program is updated and it can detect the newest malicious programs. Without this update mechanism, the anti-virus program is quite useless and doesn't give much protection against malware threats.

The mobile device users should be advised not to install any unknown or suspicious files to the device. However, there are always people who are curious to see what the unknown file or program looks like and they don't care about the security advices. For example people have been told for five years not to click unknown e-mail attachments, and still the e-mail worms are one of the most common malware types on PCs [80]. But still, raising awareness about security threats among the users is important and shouldn't be undervalued. At least this way the amount of infected devices and damage caused by malware is minimized.

7.8 Protection against Bluetooth attacks

By putting the device into non-discoverable mode gives some protection against the Bluetooth attacks that were discussed in this project. This way the attacker has to know the exact Bluetooth device address. However, there exist some tools available from the Internet that claim that they can find Bluetooth devices that are in non-discoverable mode. For example the RedFang tool brute force guesses the last six bytes of the Bluetooth device address [65]. One way to be sure, that the device is safe from attacks is to turn off Bluetooth when it isn't needed.

If the user has a vulnerable phone it should be taken to service point and its software version should be updated. At least Nokia and Sony Ericsson have released software updates to vulnerable phone models to patch the Bluetooth security holes [51, 52]. After these updates, the

phone should be safe from these attacks. Also when pairing with some Bluetooth device, a long decimal number PIN code should be used. Preferably it should be over 8 digits. This way it takes longer to brute force guess it. The attacker could monitor and record activities that happen in radio waves when pairing is done. If the PIN code is over 8 digits long, it will take years to figure out the right PIN code (>10 == over 100 million different possibilities). If the PIN code is only 4 digits long, the correct PIN code could be determined in a matter of few hours (10 == only 10 thousand different possibilities). Also if the other party is unknown, the pairing should not be made with it. Pairing should only be done with known devices. [66]

7.9 Protection against WLAN attacks

Like in PCs, a firewall should be used in mobile devices also. The firewall program inspects all network traffic and let only wanted traffic to go trough it. It is necessary to prevent the incoming unauthorized requests to device's services and the unauthorized outgoing network traffic that some malicious programs may send. The firewall also offers the possibility to monitor the network traffic. It is possible to implement event logging and alert functions to the firewall. However, the firewall cannot give protection against security holes in services that are allowed by the firewall (for example wwwserver).

A Virtual Private Network program can be used to encrypt the traffic that is transferred through the Internet. This is used to give protection against a hacker that could be eavesdropping the connection between for finding out the contents of traffic. The end point where the encrypted traffic is decrypted can be in the organizations firewall where for example the e-mail or intranet can be accessed. This kind of solution is also called tunneling, because there is a secure "tunnel" trough the Internet and the sensitive data is transferred inside this tunnel. Commonly used VPN protocols are IPSec, L2TP, and PPTP. A VPN can be used to secure all traffic that is transferred over un-secure network.

When WLAN base stations are taken into use, it is important to remember to change the default settings in them. Usually they have very low security level, if none, to ease interoperability when

they are taken into use. They might have a default password, use SSID name *ANY*, and accept unencrypted traffic.

Also the local WLAN traffic between the base station and the mobile device can be encrypted. The WEP encryption is known to be weak, but it should be used if other stronger encryption (such as WPA) methods aren't available. VPN can be also used in addition to provide more protection against possible eavesdroppers and to compensate the security vulnerabilities in WEP.

7.10 Software updates

Even if the security aspects were carefully thought in product development phase, some security holes could still be left in the system. It is very difficult, perhaps even impossible to design a system that has perfect security. The more complex system, the harder it is to design good security for it. The odds are that there are more security holes in a complex system than in a simple one. There are always people who spend their time in searching for possible security holes in systems.

For this reason, it is important observe what happens in the hacker world. When there are new security vulnerabilities discovered the response should be quick to fix the security holes in the system. After the fix is done, the new software should be delivered to users as quickly as possible and without causing too much inconvenience to users. Also the users should be notified that there is a new software version available that fixes the security holes.

Currently the software updates to mobile phones are available from authorized service points. However, this requires the users physically take the devices to service points to get updated. This is too troublesome method for some people and they leave their devices vulnerable for a long time. More sophisticated software update methods are needed to get the new software to users faster and more easily.

One method could be that the software update is delivered over the cellular network. This is called OTA (Over-The-Air) method. The operator or enterprise could manage the users mobile

devices. When there is a new software version available, the operator could send the software update to user's device via OTA.

Other method could be that the software updates are delivered via Internet. The users could download them for example from device manufacturer's website. This would be similar to Microsoft Windows update method in PCs, where it has been made quite easy for the user to download and install software patches that fix the security holes in the operating system. The users could download the updates first to PC and from there install them to the mobile device. Alternative could be that the updates are downloaded directly to mobile devices. While the mobile devices are advancing and the network speeds are increasing, the devices could also have a feature that they would automatically check from the Internet on device manufacturers or operators site if there were new updates available.

Whichever update method is used, it is necessary to have some kind of verification that the software update is really from the device manufacturer or from some other authorized source. Otherwise there would be a possibility that the hackers could slip their own piece of malicious software into the devices.

7.11 Operator security boundary

The mobile devices have very limited processing power and memory capacity. If some heavy firewall application is running on the mobile device, that is checking all the incoming and outgoing traffic, it may considerably slow down the device's performance. To ease the burden from mobile device, the data traffic could be checked on operator's or service provider's side instead.

7.12 Security policies

The security policy provides a framework for designing countermeasures against the threats. With the help of security policy it can be decided for example that is a firewall needed, how it

should be configured, are some access tokens needed and so on. It also helps the people who are in charge of security to have a consistent view on how the system should be protected against the security threats. Before creating a security policy it must be defined what things are to be protected and what are the threats. All possible threats should be imagined: what they are, how could they happen, and what are the costs of the damage. To chart the risks the following formula can be used:

$$R = C \times P$$

Where R is the risk, C is the cost of the damage, and P is the probability of the damage. After this each threat is evaluated and decided if some actions are to be taken with it.

It is very important to have some kind of security policy in a company or in an institute for its computer network and mobile devices. The policy should outline at least who is responsible for what (implementation, enforcement, audit, review…), what the basic security policies are, and why they are there. If the policies aren't explained clearly enough they might not be followed. Usually the problem is that the policies aren't followed instead of the security policies being somehow incomplete.

8 Practical implemetation (private part)

Private part of this thesis solves Nokia security architecture and test sequences which are used to make production more effective. The main criterion was to decrease test time. To make the test sequences more effective by decreasing test time we used two kinds of main methods. The first one consists in parallel implementation of the separate test steps. This method largely depend on the equipment which is used in production tests e.g. test robot, measuring equipment, signal generator. In some tests their parallel execution is not possible because they are not supported by software. The second method is based on statistic utilization. When the amount of devices which are tested is couple of thousand per day, it is clearly possible to see how big the percentage of fail tests is. Production testing is divided in to three main parts which are introduced in supporting documents for master's thesis. First of this part is Flashing and Alignment. The purpose of the Flashing and Alignment (FLALI) test phase is to check that the phone does not draw excessive or no current, flash the MCU software but only in the case that Panel Flash MCU flashing was not successful so the phone can be tested. Very important is also to check and calibrate the base band energy management system and calibrate the RF parameters.

Second test phase is Final User Interface testing. The purpose of the Final User Interface testing (FINUI) is to test the base band and RF alignments, if the LCD backlights, contrast and display are functioning correctly, if the keypad backlights are functioning, if the keys depress and return to their non-operating position and the audio functionality and quality. In this test phase is test station generating all kind of calls and tying to connect the phone. All values are measured and evaluate. The number of test depends of kind of phone and equipment. Each kind of phone has own adapter. This phase is the only one where is no flashing files to the phone.

Labeling is last of the testing phases using in production. It is a phase where a terminal gets its unique International Mobile Equipment Identity (IMEI). IMEI can be electronic Serial Numbers (ESN) or Mobile Equipment Identifier (MEID) type. Content (CP) and Language (PPM) packages are also flashing to the phone in this phase.

This material contains company confidential materials and it is introduced in Supporting documents for master's thesis (see appendix A).

9 Summary

This thesis presented different security threats related to mobile devices. Different malicious program types were introduced. There already exists malware written for mobile devices. Most of these programs, that have emerged, are proof-of-concept type programs. They don't do much harm and require user interaction to get installed to the device. However, in most cases, their source codes have been released on the Internet. It can be guessed that new malware and their variants will probably emerge for mobile devices in the near future. There is also a risk that malware, which exploits the victim financially, will emerge for mobile devices.

Some of the Bluetooth attacks that were described in this thesis are quite severe. However these attacks work only on some phone models. The device manufacturers have recently released software updates to vulnerable phone models. The users that have these vulnerable models should take them to authorized service point and update the phone's software. After this, these attacks shouldn't work anymore against these phones.

The WEP encryption that is used in WLAN is weak. In highly trafficked network that has WEP encryption, the secret key can be determined with a special program in few hours. These programs are available in the Internet. A more powerful WPA encryption has been developed to replace the weak WEP. The users of WLAN devices have to be careful when logging into unknown WLAN access points for example in airports. The hackers may have established their own access points to collect credit card information and passwords. The users need to be careful always when this kind of information is asked.

The objects that were set to this thesis were achieved. Many security threats were found. Also some preventive measures were studied.

There are many possibilities how this research work could be continued. On the software implementation level, it would be interesting to study buffer overflow attack methods and

how they could be prevented. The WLAN security threats could be examined further. Also it would be important to study further what measures could be taken in product development to improve the security of the devices. In general, the whole security area is a very large and interesting subject to be studied. New information about different security threats is released almost daily in the Internet.

Bibliography

[1] Digia Inc., *Programming for the Series 60 Platform and Symbian OS*, John Wiley & Sons Ltd, 2003.

[3] Stallings William, *Network security essentials: Applications and standards, Second Edition*, Prentice-Hall, 2003.

[4] Podrezov Alexey, F-Secure Corporation, *F-Secure virus descriptions: Backdoor,* [www-document], http://www.f-secure.com/v-descs/backdoor.shtml.

[5] Podrezov Alexey, F-Secure Corporation, *F-Secure virus descriptions: Trojan*, [www-document], http://www.f-secure.com/v-descs/trojan.shtml.

[6] Niemelä Jarno, Rautiainen Sami, Tocheva Katrin, F-Secure Corporation, *F-Secure virus descriptions: Cabir*, [www-document], http://www.f-secure.com/v-descs/cabir.shtml.

[7] Laurie Adam, Holtmann Marcel, Herfurt Martin, *Hacking Bluetooth enabled mobile phones and beyond – Full Disclosure*, Lecture slides from 21st Chaos Communication Congress, December 27th to 29th 2004, Berlin, [pdf-document], http://www.ccc.de/congress/2004/fahrplan/files/169-bluetooth-hacking-slides.pdf

[8] Niemelä Jarno, F-Secure Corporation, *F-Secure virus descriptions: Skulls*, [www-document], http://www.f-secure.com/v-descs/skulls.shtml.

[9] Laurie Adam, Laurie Ben, A.L. Digital Ltd., *Serious flaws in Bluetooth security lead to disclosure of personal data*, [www-document], http://www.thebunker.net/security/bluetooth.htm.

[10] Blau John, IDG news service, *New Trojan threatens Symbian smart phones*, [www-document], http://www.computerworld.com/securitytopics/security/virus/story/0,10801,98515,00.htm

[11] Niemelä Jarno, F-Secure Corporation, *F-Secure virus description: MGDropper*, [www-document], http://www.f-secure.com/v-descs/mgdropper.shtml.

[12] Niemelä Jarno, F-Secure Corporation, *F-Secure virus description: Cabir.H*, [www-document], http://www.f-secure.com/v-descs/cabir_h.shtml .

[13] F-Secure Corporation, *Evolution in Cabir variants,* [www-document], http://www.f-secure.com/weblog/archives/archive-122004.html#00000414

[14] Niemelä Jarno, F-Secure Corporation, *F-Secure virus description: Mquito*, [www-document], http://www.f-secure.com/v-descs/mquito.shtml

[15] Symbian Ltd, *Information about Mosquitos Trojan,* [www-document], http://www.symbian.com/press-office/2004/pr040810.html

[16] Bray Jennifer, Sturman Charles F, *Bluetooth, Connect Without Cables, Second Edition,* Prentice Hall PTR, 2002.

[17] Niemelä Jarno, F-Secure Corporation, *F-Secure virus description: Skulls.B*, [www-document], http://www.f-secure.com/v-descs/skulls_b.shtml

[18] Niemelä Jarno, F-Secure Corporation, *F-Secure virus description: Skulls.C*, [www-document], http://www.f-secure.com/v-descs/skulls_c.shtml.

[19] F-Secure Corporation, *Improved disinfection instructions for Cabir,* [www-document], http://www.f-secure.com/weblog/archives/archive-112004.html#00000366

[20] *Mobile Ad-hoc Viruses,* [www-document], http://www.mobilecommunitydesign.com/archives/000088.php

[21] Erdelyi Gergely, F-Secure Corporation, *F-Secure virus descriptions: Duts.1520,* [www-document], http://www.f-secure.com/v-descs/dtus.shtml

[22] Hyppönen Mikko, Niemelä Jarno, F-Secure Corporation, *F-Secure virus descriptions: Brador,* [www-document], http://www.f-secure.com/v-descs/brador.shtml.

[24] Podrezov Alexey, F-Secure Corporation, *F-Secure virus descriptions: Constructor,* [www-document], http://www.f-secure.com/v-descs/virmaker.shtml.

[25] Peikari Cyrus, Fogie Seth, Ratter/29A, *Details emerge on the first Windows Mobile virus,* [www-document], http://www.informit.com/articles/article.asp?p=337069

[26] F-Secure Corporation, *F-Secure virus descriptions: EICAR-test,* [wwwdocument], http://www.f-secure.com/v-descs/eicar.shtml

[27] Niemelä Jarno, F-Secure Corporation, *F-Secure virus descriptions: Lasco.A,* [www-document], http://www.f-secure.com/v-descs/lasco_a.shtml.

[28] Peikari Cyrus, Fogie Seth, Ratter/29A, Read Jonathan, *Reverse-Engineering the first Pocket PC Trojan, Part 2,* [www-document], http://www.informit.com/articles/article.asp?p=340545&seqNum=3

[29] Canalys, *Global smart phone shipments treble in Q3,* [www-document], http://www.canalys.com/pr/2004/r2004102.htm.

[31] Bluetooth SIG, *Bluetooth SIG promoter members,* [www-document], http://www.bluetooth.com/about/promoter.asp.

[32] Miller Brent A., Bisdikian Chatschik, Ph.D., *Bluetooth revealed, second edition,* Prentice Hall PTR, 2002.

[90] Peikari Cyrus, Fogie Seth, *Cracking WEP*, [pdf-document], http://www.airscanner.com/pubs/wep.pdf.

[34] *Antennas, Amplifiers and Propagation Topics (for Microwave WLANs)*, [www-document], http://www.wlan.org.uk/antenna-page.html.

[35] Nokia, *Press photos, Nokia 9500 phone*, [picture], http://www.nokia.com/press/photo/view.html?imgURL=/press/photo/phones/jpeg/9500_open_lores.jpg.

[36] Sony Ericsson, *P802 Photo Library*, [picture], http://www.sonyericsson.com/downloads/P800_4.zip.

[37] Nokia, *Press photos, Nokia 6630 phone*, [picture], http://www.nokia.com/press/photo/view.html?imgURL=/press/photo/phones/jpeg/34_6630.jpg.

[38] Nokia, *Platform architectures: Series 80*, [www-document], http://www.nokia.com/nokia/0,,62622,00.html.

[39] *Series 60 Platform technical info*, [www-document], http://www.series60.com/technicalinfo.

[40] Symbian Ltd., *Company ownership*, [www-document], http://www.symbian.com/about/ownership.html.

[42] Microsoft Corp., *Windows CE*, [www-document], http://msdn.microsoft.com/embedded/getstart/choose/whywinemb/wince/default.aspx.

[43] Evers Joris, IDG News Service, *Microsoft to phase out Pocket PC, Smartphone brands*, [www-document], http://www.infoworld.com/article/05/01/06/HNmicrosoftphaseoutpocket_1.html .

[44] Microsoft Corp., *Frequently asked questions about Windows Mobile Software*, [www-document], http://www.microsoft.com/windowsmobile/about/faq.mspx.

[45] Microsoft Corp., *QTEC 8010*, [www-document], http://www.microsoft.com/windowsmobile/devices/devicedisplay.aspx?module=deviceDisplay;Smartphone;emea;140.

[46] Microsoft Corp., *HP iPAQ h4150/h4155*, [www-document], http://www.microsoft.com/windowsmobile/devices/devicedisplay.aspx?module=deviceDisplay;PPC;emea;74.

[47] Microsoft Corp., *Compare devices*, [www-document], http://www.microsoft.com/windowsmobile/devices/compare.mspx.

[48] Webopedia, *What is snarf? – A Word Definition From The Webopedia Computer Dictionary*, [www-document],http://www.webopedia.com/TERM/S/snarf.html.

[49] Gehrmann Christian, Persson Joakim, Smeets Ben, *Bluetooth Security,* ARTECH HOUSE INC., 2004.

[50] *BlueBug*, [www-document], http://trifinite.org/trifinite_stuff_bluebug.html.

[51] Nokia, *Bluetooth Security*, [www-document], http://www.nokia.com/nokia/0,,65909,00.html.

[52] Sony Ericsson, *Bluetooth security,* [www-document], http://www.sonyericsson.com/spg.jsp?cc=us&lc=en&ver=4000&template =pp1_loader&php=fhp1_5_2730&zone=pp&lm=pp1&pid= .

[53] Herfurt Martin, Mulliner Collin, *Blueprinting – Remote Device Identification Based On Bluetooth Fingerprinting Techniques*, [pdf document], http://trifinite.org/Downloads/Blueprinting.pdf.

[54] MacDonnell Ulsch, *Security strategies for E-Companies,* [wwwdocument], http://infosecuritymag.techtarget.com/articles/july00/columns2_ec_doesit.shtml.

[55] Gaudin Sharon, Network World 03/04/02, *Net saboteur faces 41 months*, [www-document], http://www.nwfusion.com/news/2002/0304lloyd.html.

[57] Wikipedia, *Botnet,* [www-document], http://en.wikipedia.org/wiki/Botnet.

[58] *BlueSmack,* [www-document], http://trifinite.org/trifinite_stuff_bluesmack.html.

[59] *BlueZ – Official Linux Bluetooth protocol stack*, [www-document], http://www.bluez.org.

[60] *Long distance snarf,* [www-document], http://trifinite.org/trifinite_stuff_lds.html.

[61] *1 Mile Bluesnarf on The Screen Savers*, [video clip], http://www.socalwug.org/media/1mile-bluetooth-g4techtv-091604.rm.

[62] *Antennas*, [www-document], http://trifinite.org/trifinite_links.html.

[63] *Bluetooone*, [www-document], http://trifinite.org/trifinite_stuff_bluetooone.html.

[64] *Blooover*, [www-document], http://trifinite.org/trifinite_stuff_blooover.html.

[65] *RedFang*, *Bluetooth Discovery Tool*, [www-document], http://www.securiteam.com/tools/5JP0I1FAAE.html.

[66] Bluetooth SIG, *Wireless security*, [www-document], http://www.bluetooth.com/help/security.asp.

[68] Federal Bureau of Investigation, *Wanted by the FBI, Computer intrusion, Saad Echouafni*, [www-document], http://www.fbi.gov/mostwant/fugitive/jan2005/janechouafni.htm.

[69] HP, *HP Over-the-Air Management Solution Drives Handset Personalization, Enhances Customer Support Services,* [pdf-document], http://www.hp.com/hpinfo/newsroom/press_kits/2005/3gsm/fs_overtheair.pdf.

[71] F-Secure Corporation, *F-Secure Corporation's Data Security Summary for 2006*, [www-document], http://www.f-secure.com/2006/ .

[73] Anti-Phishing Working Group, *Phishing Activity Trends Report December 2004*, [pdf-document], http://antiphishing.org/APWG%20Phishing%20Activity%20Report %20-%20December%202004.pdf.

[74] F-Secure Corporation, *Outrageous Red Cross scam site taken down*, [www-document], http://www.f-secure.com/weblog/archives/archive-012005.html#00000438.

[75] Potter Bruce, Caswell Brian, *Bluesniff – The Next Wardriving Frontier*, Presentation material from DEFCON 11, August 1st to 3rd 2003, Las Vegas, [PowerPoint document], http://www.defcon.org/images/defcon-11/dc-11-presentations/dc-11-Potter/dc-11-potter.ppt.

[76] Pointsec Mobile Technologies AB, *Stolen PDAs Provide Open Door To Corporate Networks*, [pdf-document], http://www.pointsec.com/news/download/PointsecNo3-September2003.pdf.

[77] Siezen Sander, Symbian Ltd., *Symbian OS Version 9.5, Product Description, Revision 1.1*, http://www.symbian.com/technology/SymbianOS_v9_prod_descv1.pdf.

[78] NewLC, *Introduction to Symbian OS v9*, [www-document], http://www.newlc.com/article.php3?id_article=742.

[79] Symbian Software Ltd, *Symbian Signed*, [www-document], https://www.symbiansigned.com/app/page/faq.

[80] F-Secure Corp., *More Cabir sightings*, [www-document], http://www.fsecure. com/weblog/archives/archive-022005.html#00000481.

[81] Roudnev Alexei, *Site elimination service – I received offer by spam*, [www-document], http://merit.edu/mail.archives/nanog/2004-06/msg00089.html.

[82] Schneier Bruce, *Secrets and Lies, Digital security in a networked world*, Wiley Publishing Inc., 2000.

[83] Newitz Annalee, Wired magazine, *The Great Cell Phone Robbery*, [wwwdocument], http://www.wired.com/wired/archive/12.12/phreakers.html?pg=4.

[84] Yamada Kotaro, Aoshima Ken, *Pranksters dial i-mode crazy*, [wwwdocument], http://lists.virus.org/isn-0007/msg00119.html.

[85] *Zelos Group predicts Smartphone takeover*, [www-document], http://www.mobilemag.com/content/100/344/C2408/.

[86] Poulsen Kevin, SecurityFocus, *Hacker penetrates T-Mobile systems*, [www-document], http://www.securityfocus.com/news/10271.

[88] Geier Jim, *Wireless LANs, Second Edition*, Sams Publishing, 2002.

[90] Peikari Cyrus, Fogie Seth, *Cracking WEP*, [pdf-document], http://www.airscanner.com/pubs/wep.pdf.

[91] F-Secure Corp., *More From DEFCON*, [www-document], http://www.fsecure. com/weblog/archives/archive-082004.html#00000247.

[92] *Antennas, Amplifiers and Propagation Topics (for Microwave WLANs)*, [www-document], http://www.wlan.org.uk/antenna-page.html.

Appendix

Apendix A. Supporting documents for master’s thesis