NTNU
Norwegian University of
Science and Technology

Security as a Service in Cloud for Smartphones

lakshmi subramanian

Master in Security and Mobile Computing
Submission date: June 2011
Supervisor: Danilo Gligoroski, ITEM
Co-supervisor: Philipp Stephanow, Fraunnhofer SIT, Munich, Germany
Gerald Q.Maguire Jr, Royal Institute of Technology
(KTH),Stockholm

Norwegian University of Science and Technology
Department of Telematics

Problem Description

Lakshmi Subramanian

The mobile computing paradigm has seen tremendous advancements in recent times. Smartphones have emerged as a type of mobile device providing “all-in-one” convenience by integrating traditional mobile phone functionality and the functionality of handheld computers. Today smartphones offer PC-like functionality to end users allowing them to check their e-mail, maintain calendars, browse the internet, watch videos, play music, etc. In addition to these functions, they are also used for privacy sensitive tasks such as on-line-banking - these tasks make them an attractive platform for attackers. Enormous numbers of applications are being developed for each of the mobile operating systems (OSs) and each application has its own security requirements (and vulnerabilities).

Storing personal data on the smartphone has become a common practice. Awareness of the risks associated with smartphone usage is relatively low when compared to the awareness of risks for desktop computers. Sensitive data such as email and bank passwords are frequently stored by users in an unsafe manner on their smartphones. Therefore, there is a growing need to address the security risks associated with smartphones.

Although there seems to be significant developments in terms of available computing power, local storage, and other capabilities of smartphones in comparison to so called “feature phones”, desktop computing devices have evolved to a much greater extent – especially with respect to security.

In a corporate organisation set up, sensitive corporate data is stored by each employee of the company. With the use of smartphones, the tendency to use a smartphone for official purposes is also on the rise as it is quite handy. For example, employees carrying mobile phones to meetings instead of using laptops. Therefore, it becomes highly important to protect the information from being disclosed and misused by external entities. Furthermore, it becomes necessary to ensure that the employees abide by the policies of the company to ensure security.

In short, this project addresses this problem by proposing a generic architecture for providing security services in the cloud for smartphones

Assignment given: 25.01.2011

Supervisor: Prof. Danilo Gligoroski

Industry Supervisor: Philipp Stephanow, Fraunhofer SIT

Abstract

Smartphone usage has been continuously growing in recent times. Smartphones offer Personal Computer (PC) functionality to the end user, hence they are vulnerable to the same sorts of security threats as desktop computers. Cloud computing is a new computing paradigm and a breakthrough technology of recent times. Its growing popularity can be attributed to its ability to transform computing to a utility, scalability, and cost effectiveness. More and more services are predicted to be offered in the cloud in the near future.

Due to the resource constraints of smartphones, security services in the form of a cloud offering seems to be a natural fit (as the services could be provided in a very scalable form in the cloud while off-loading the smartphone).

This project proposes a generic architecture for providing security services in the cloud for smartphones. To enable the design of this architecture, it is essential to analyze and identify possible security solutions that could be provided as a cloud service to the smartphone. Security requirements of smartphones have been analysed considering the various infection channels for smartphones, attacks and threats encountered in a smartphone environment, smartphone usage scenarios and the smartphone's limitations. Next, the security functions that must be implemented in the smartphone to overcome these threats are identified. Furthermore, a review of the existing architectures for mobile computing are presented and their security issues are examined.

A detailed study of the analysed results has been used to build the architecture for offering security services to smartphones in the cloud, targeted use case scenario being the usage in a corporate environment. The functions to be handled by each of the components of the architecture have been specified. Furthermore, the proposed architecture has been examined to prove its feasibility by analysing it in terms of its security aspects, scalability and flexibility. Additionally, experiments to understand the performance enhancement by offering security services in the cloud for smartphones have been performed. This has been done by measuring the resource consumption of anti-virus software in a smartphone and performing the same measurement in an emulated smartphone in the cloud.

Dedicated to my family and friends.

Acknowledgements

My heartfelt gratitude to my thesis advisor Mr. Philipp Stephanow (Dept. of Secure Services and Quality Testing, Fraunhofer Institute for Secure Information Technology (SIT), Munich, Germany) for sharing his valuable ideas and also personally helping me settle down in a new country and successfully complete my work in a very short span.

My sincere thanks are due to my thesis supervisor Prof. Dr. Gerald Q. “Chip”Maguire Jr. (School of Information and Communication Technologies, Royal Institute of Technology (KTH), Stockholm, Sweden) for his valuable suggestions, thought provoking ideas and indispensable recommendations. I am very grateful for his spending valuable time in guiding me and getting back to me in a very short span whenever approached.

Special Thanks to Mr. Tobias Wahl (Dept. of Secure Services and Quality Testing, Fraunhofer SIT) for helping me during the initial stages of the project.

I would also take this opportunity to appreciate the efforts of Prof. Dr. Danilo Gligoroski (Dept. of Telematics, Norwegian University of Science and Technology (NTNU), Trondheim, Norway) for his critical reviews on my work and constant encouragement. Loads of thanks are also due to my friend Sathya for constructive comments on the thesis.

My gratitude to Ms. Mona Nordaune (NordSecMob Co-ordinator, NTNU) and Ms. May-Britt Eklund-Larsson (NordSecMob Co-ordinator, KTH) for being very helpful during my Erasmus Mundus days in Norway and Sweden respectively. I also appreciate the constant support of the group members of the Dept. of Secure Services and Quality Testing at Fraunhofer SIT.

I would like to thank Fraunofer SIT for providing me with an opportunity to pursue my master’s thesis in their facility. Lastly, my gratitude to the European Commission for offering me the Erasmus Mundus grant and to all my fellow course mates for making my time a memorable one during the last couple of years. Without the assistance I have received from these people, my individual inquiry would have been much more difficult, and the experience much less rewarding.

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

3DES	Triple Data Encryption Standard
3G	Third Generation
AES	Advanced Encryption Standard
AWS	Amazon Web Services
BES	BlackBerry Enterprise Server
BIOS	Basic Input/Output System
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DDoS	Distributed Denial of Service
DoS	Denial of Service
EDGE	Enhanced Data rates for GSM Evolution
ENISA	European Network and Information Security Agency
GB	Gigabyte
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communications (originally “Groupe Spécial Mobile”)
HD	High Definition
HDMI	High Definition Multimedia Interface
HMAC	Hash-based Message Authentication Code
HTML	Hypertext Markup Language
ID	Identifier
IP	Internet protocol
IT	Information Technology
IaaS	Infrastructure as a Service
JRE	Java Runtime Environment
JVM	Java Virtual Machine
MMC	Multimedia Card
MMS	Multimedia Message System
MTM	Mobile Trusted Module
mTAN	mobile Transaction Authentication Number
NIST	(U. S.) National Institute of Standards and Technology
NDA	Non Disclosure Agreement
OS	Operating System
PC	Personal Computer

PDA	Personal Digital Assistant
PGP	Pretty Good Privacy
PKI	Public Key Infrastructure
PaaS	Platform as a Service
RC4	Rivest chipher 4
RIM	Research in Motion
RSA	Rivest Shamir and Aldeman
S/MIME	Secure Multipurpose Internet Mail Extension
SD	Secure digital
SDK	Software Development Kit
SHA	Secure Hash Algorithm
SMS	Short Message Service
SOA	Service Oriented Architecture
SSH	Secure Shell
SSL	Secure Socket layer
SaaS	Software as a Service
SeaaS	Security as a Service
SxC	Security by Contract
TCP/IP	Transmission Control Protocol/Internet protocol
TPM	Trusted Platform Module
U.S.	United States (of America)
VM	Virtual Machine
VPC	Virtual Private Cloud
WAP	Wireless Application Protocol
WS	Web Service
Wi-Fi	Wireless Fidelity (a tradename for IEEE 802.11 compliant devices)

1 Introduction

1.1 Overview

Over the last decade, the popularity of handheld devices such as Personal Digital Assistants (PDAs) and smartphones have increased tremendously. Gartner forecasts that the number of smartphones will exceed the number of Personal Computers (PCs) by 2013 [1]. Estimates suggest that the United States (U.S) smartphones sales are expected to grow from 67 million units in 2010 to 97 million units in 2011 [1]. With this tremendous potential growth in the numbers of smartphones, concerns about the security of these mobile phones are also on the rise. Rich personal data and/or corporate data are increasingly stored in smartphones. In most cases there is little concern being given to the security of this information. Cisco's annual Internet security threat report predicts that criminals are already targeting smartphones, rather than traditional Microsoft Windows PCs [3]. This report also predicts that during 2011 we will see an increasing number of attacks directed against these devices. The resource constraints of these devices seem to be a major limiting factor preventing them from supporting more powerful security services (See section 3.5 where the limitations of smartphones in terms of providing security services on the device is discussed). Offloading computationally intensive security services to the cloud could be extremely beneficial for users of these smartphones.

The simplicity and scalability that cloud computing offers has attracted the attention of both users and organisations. The United States Federal IT Market forecasts cloud computing as one of the technology segments that will witness double digit growth between 2011 and 2015 [4]. The application of cloud computing in mobile phones has caught the attention of researchers worldwide as there is a good match between these resource constrained handheld devices and the resource abundant cloud. (The work done by various researchers in the field of mobile cloud computing is presented in the related work section 2.5.)

1.2 Problem Description

The mobile computing paradigm has seen tremendous advancements in recent times. Smartphones have emerged as a type of mobile device providing “all-in-one” convenience by integrating traditional mobile phone functionality and the functionality of handheld computers. Various models of smartphones have been released catering to the various demands of mobile users. Today smartphones offer PC-like functionality to end users allowing them to check their e-mail, maintain calendars, browse the internet, watch videos, play music, etc. In addition to these functions, they are also used for privacy sensitive tasks such as on-line-banking - these tasks make them an attractive platform for attackers. Enormous numbers of applications are being developed for each of the mobile operating systems (OSs) and each application has its own security requirements (and vulnerabilities). Heterogeneity in hardware, software, and communication protocols to connect to the Internet for all of the different smartphones add complexity when attempting to definite security functions for smartphones. This heterogeneity also increases the difficulty in designing, implementing, and testing applications for these smartphones.

Storing personal data on the smartphone has become a common practice. Awareness of the risks associated with smartphone usage is relatively low when compared to the awareness of risks for desktop computers. Sensitive data such as email and bank passwords are frequently stored by users in an unsafe manner on their smartphones. These poor security practices attract attackers to concentrate on smartphone platforms in order to exploit the vulnerabilities of the smartphone OSs and application software, as well as user generated vulnerabilities. Therefore, there is a growing need to address the security risks associated with smartphones.

Although there seems to be significant developments in terms of available computing power, local storage, and other capabilities of smartphones in comparison to so called “feature phones”, desktop computing devices have evolved to a much greater extent – especially with respect to security. Part of the reason for this may be that desktop computers have been programmable by users for many decades, while only in recent years has it been possible for more than a very small and carefully controlled group of developers to create software for a mobile phone.

Offloading computation from resource constrained devices has been an area of focus for researchers. This aim of this thesis project is to improve the perceived performance of mobile devices by utilizing the broadband wireless connectivity of these devices. Security functions such as anti-virus scanning are resource intensive and additionally this computation and associated memory activity will deplete the battery power of the smartphone. Cloud computing seems to be a good fit by shifting the computation from the mobile devices to the cloud, hence exploiting the computational power of the cloud and the fact that the cloud computers are provided with mains power. This suggests that if there are computationally intensive security services that can be migrated to the cloud, then Security as a Service (SeaaS) for smartphones is one way in which improved security could be offered as a service in the cloud for the users of smartphones.

In a corporate organisation set up, sensitive corporate data is stored by each employee of the company. With the use of smartphones, the tendency to use a smartphone for official purposes is also on the rise as it is quite handy. For example, carrying mobile phones to meetings instead of using laptops. Therefore, it becomes highly important to protect the information from being disclosed and misused by external entities. Furthermore, it becomes necessary to ensure that the employees abide by the policies of the company to ensure security.

Chapter 2 discusses the general background information required for the rest of the thesis. Chapter 3 deals with some of the major security aspects of a smartphone. Chapter 4 elaborates some of the key architectures that have been identified to realise a secure architecture for the smartphones. Chapter 5 gives a detailed description of the proposed architecture. Chapter 6 describes the performance measurements for anti-virus scanning. Chapter 7 analyses the proposed architecture for the various security aspects. Chapter 8 presents the conclusions and suggests some future work.

2 General Background

This chapter focuses on providing a general background about smartphones (section 2.1) and cloud computing technology (section 2.2) so that later, it will be easier to understand how and why they are related. The service and deployment models of cloud computing are discussed in sections 2.3 and 2.4 respectively. At the end of this chapter, a short description of the work done by other researchers in the field of mobile cloud computing and smartphone security is presented in section 2.5.

2.1 Smartphones

Smartphones are a category of mobile phones which are “smart” (i.e., more capable) when compared to traditional mobile phones. Smartphones are targeted to address the need for a pocket PC in addition to a phone. As a result they offer many features which are not usually associated with mobile phones, such as the ability to run downloaded software applications, web browsing capabilities, etc. Figure 1 contrasts the difference in style of a traditional mobile phone and a smartphone with regard to their appearance. Two of the most obvious differences are the larger screen and keyboard of the smartphone – as interaction is no longer limited to browsing menus, dialling phone numbers, storing/retrieving phone book entries, entering/reading short messages, and playing simple built-in standalone games. These are some key features of smartphones which make it “smarter” than traditional mobile phones. The following subsections will examine some of the features which characterize smartphones.

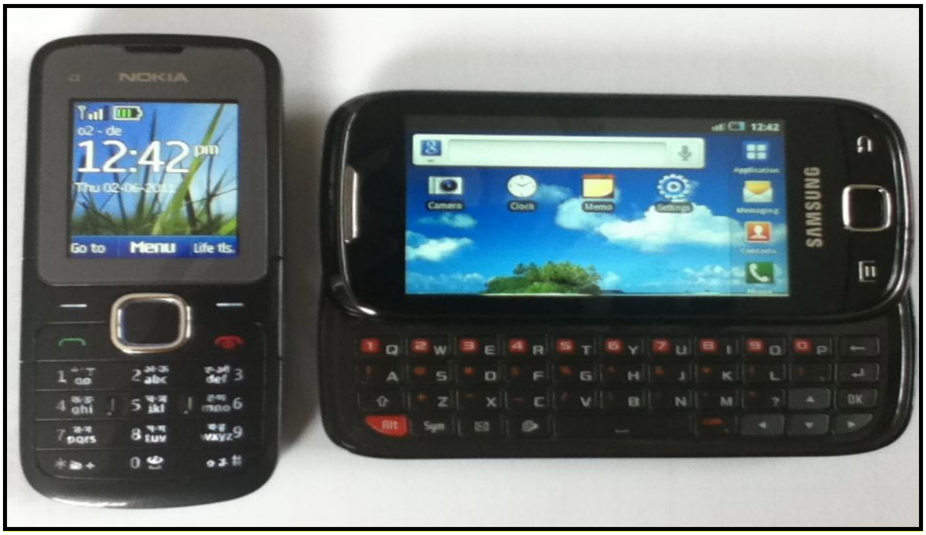


Figure 1: Comparison between a traditional mobile phone and a smartphone

2.1.1 Operating System as a platform for applications

One of the key features of smartphones is that user can install new applications on the phone, rather than just run an application in a Java Virtual Machine (JVM). To enable local applications requires that the applications be able to call on the services of an operating system. Unlike traditional mobile phones, smartphones have general purpose OSs, examples include: the Research in Motion’s (RIM) BlackBerry OS, Apple’s iPhone iOS, Google’s Android OS, Microsoft’s Windows Mobile, Nokia’s Symbian OS, and Linux. Based on the nature of the underlying OS, these smartphones exhibit some variations.

2.1.2 Software

Software development environments for each of the major smartphone platforms have fostered the growth of communities of software developers. The result is that a wide variety of software applications are now available and can be downloaded to the smartphone and installed by the user. In some cases these applications are distributed via a vendor specific market place, for example, Apple’s Appstore where the user can select applications for use on an iPhone. These marketplaces have more or less strict requirements and testing of applications that they distribute. Most smartphone platforms require that the software be signed (to be installed and run on a class of phones) or the user of the phone has to specifically permit the software to be installed and run.

2.1.3 Improved Internet connectivity

Today most smartphones support Internet connectivity with reasonable speed via Third Generation (3G) and/or Wireless Fidelity (Wi-Fi) technologies. As a result smartphone users can use mobile web browsers to do nearly all of the same tasks that they can do via a browser on a desktop computer. Until recently one of the exceptions was the inability to run Adobe’s Flash player on the Apple iPhone – perhaps because a full Flash player would circumvent the control which Apple has on applications via their AppStore.

2.1.4 QWERTY Keyboard

Smartphones generally come with a QWERTY keyboard. This keyboard can be either a physical keyboard or emulated via a touch screen keyboard.

2.1.5 Large amounts of local storage

The storage capacity of smartphones is generally very substantial. For example, the Android based smartphone Samsung Galaxy S II has 1 Gigabyte (GB) of Random Access Memory (RAM), 16 or 32 GB of internal storage and can support a micro Secure Digital (SD) card with upto 32 GB of storage [5]. These storage capacities are huge when compared with conventional mobile phones. This large amount of storage enables users to store their favourite audio and video files on their smartphone (rather than needing to stream this media content to the phone). Additionally or alternatively, this storage could be used for storing lots of different applications and other data.

2.1.6 Increased processing via multiple processor cores

Another trend in smartphones is the introduction of multicore processors, for example a combination of an Advanced Reduced instruction set computer (RISC) Machine (ARM) processor core and a nVidia graphics processor core – to enable very high performance video rendering. The high performance graphics capabilities are for both multimedia (for example, some of the latest smartphones have High-Definition Multimedia Interface (HDMI) output in order to drive a high definition (HD) display) and for gaming.

2.2 Cloud Computing

Cloud computing has emerged as a new computing paradigm providing hosted services by exploiting the concept of dynamically scalable and shared resources accessible over the internet. A cloud service is rented on demand, i.e. based on the customer's current requirements. Because the cloud provider can dynamically allocate virtual processors to their customers, cloud computing is highly scalable, hence the user can have as much or as little service as he or she wants at any given time. Depending on the type of the cloud service rented, the responsibility of the user in managing the service varies.

By utilizing subscription based payment for resources and services a customer can substantially reduce their operational and capital costs. Cloud computing caters to the customer's needs by offering a way to rapidly increase capacity when needed or to add new capabilities on the fly while minimizing investments in new infrastructure, training new personnel, licensing new software, etc. Figure 2 presents the U. S. National Institute of Standards and Technology (NIST) visual model of cloud computing. The following two sections describe the **service** and **deployment** models that have been proposed for cloud computing.

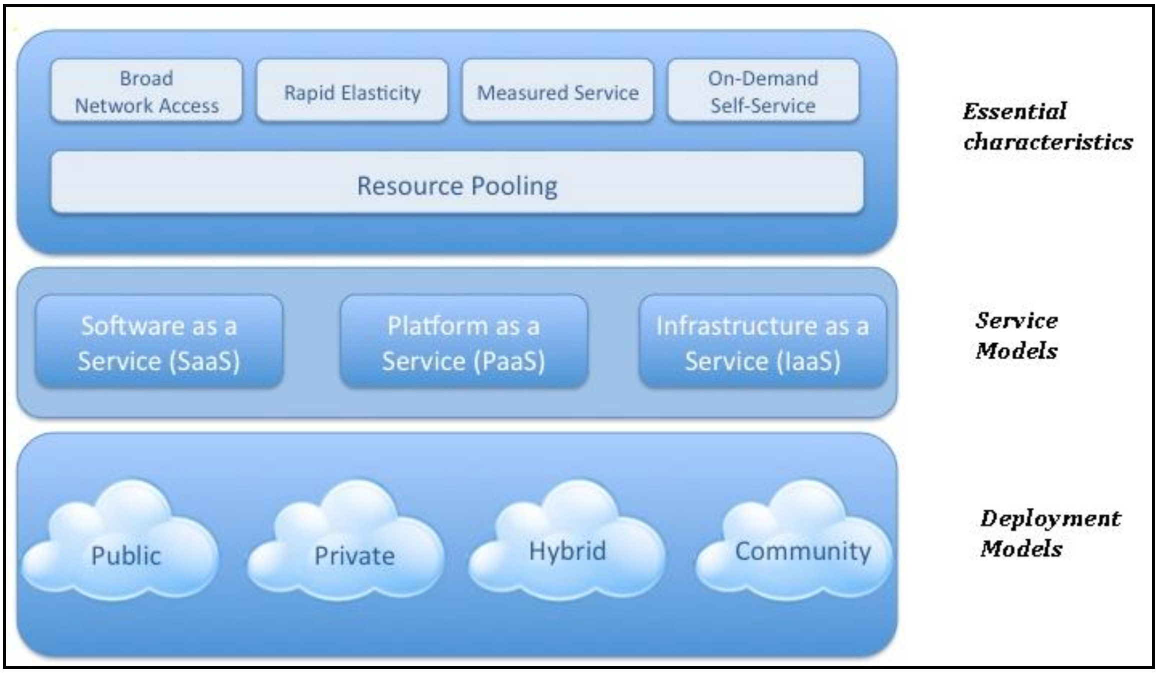


Figure 2: Cloud Computing Definition - NIST Visual model [6]

2.3 Service Models

Cloud computing can be classified based on the service model it offers, specifically: software, platform, or infrastructure. These can also be seen as a hierarchy of services, since Software as a Service (SaaS) is built on Platform as a Service (PaaS), which is in turn, built on Infrastructure as a Service (IaaS). Each of these will be described in more detail below, starting from the bottom up.

2.3.1 IaaS

In Infrastructure as a Service (IaaS), storage, computation, and network resources are the major components that are provided as a service to the customer. Customers can run their choice of operating system and other software on the infrastructure provided by the cloud provider. It is not possible for the customer to modify the physical configurations of the underlying infrastructure, although the user can request changes from the cloud provider.

2.3.2 PaaS

In the Platform as a Service (PaaS) model, the cloud provider provides a platform for developing and running the web based applications. This platform provides all the facilities to support the complete life cycle of building and delivering the applications to end users. Software and service developers are the main users of PaaS.

2.3.3 SaaS

The end user is the customer for Software as a Service (SaaS), since SaaS provides a complete software application running in the cloud. Logically underlying this are PaaS and IaaS. Generally, the customer will access the services through a web browser, but the service could also be serving end users that are “things” rather than people – for example, collecting metering information from electric meters and providing: (1) billing data to an electric supplier or distributors billing application and (2) users with access to their billing data via a web interface.

2.4 Deployment Models

Clouds can also be categorized based on the deployment model of the underlying infrastructure. The architecture of the infrastructure, location of the data centre, and specific customer requirements influence the choice of the deployment model. Note that these categories are orthogonal to the service models, thus one can have a private SaaS or a public SaaS, etc.

2.4.1 Public Cloud

A cloud service provider owns a public cloud’s physical infrastructure. This public cloud can be used to run applications for different customers. These customers share the cloud infrastructure and pay for their resource utilization based on a utility model of computing. A well designed public cloud infrastructure is designed so that each of the customers sees only their own current portion of the infrastructure.

2.4.2 Private Cloud

A pure private cloud is built exclusively for one customer who owns the infrastructure and has full control of this cloud. While such a private cloud is owned by a customer, it can be built, installed, operated, and managed by a third party - rather than the customer himself. The physical servers may be located on the customer’s premises or can be sited in a co-location facility.

The concept of a ‘virtual private cloud’ has emerged as an alternative to private clouds. In this virtual private cloud a customer is allocated a private cloud within the physical infrastructure of a public cloud. Since specific resources are allocated to this customer within the cloud, the customer may have some agreed upon assurance that this customer’s data is only stored on and only processed on dedicated servers. This approach implies that these allocated servers are **not** shared with other customers of the cloud service provider.

Sometimes the virtual private cloud customer may have an agreement that lets them expand into the public cloud when they need additional resources, it is then up to the customer to see that they do not allow critical data to be leaked via the public cloud processing. This approach enables the customer to get extra resources when they need it, but the security issues are more complex than for a pure private cloud or virtual private cloud.

2.4.3 Community Cloud

Customers having similar requirements can share infrastructure and configuration management of the cloud. As with a private cloud the management of the cloud can be done by third parties.

2.4.4 Hybrid Cloud

A combination of private and public clouds can form a hybrid cloud which can be managed by a single entity, when there is sufficient commonality in the standards used by the individual clouds of the hybrid cloud.

2.5 Related Work

This thesis project mainly focuses on offloading resource intensive security functions to the cloud, thereby providing these security functions as a service to the smartphones. Research in this area has not gained much momentum yet. Chun and Maniatis [7] present the concept of augmenting computation on a smartphone with a clone of the smartphone in the cloud. This architecture is described in greater detail in Section 4.5. Oberheide et al. [8] propose performing anti-virus scanning for smartphones in cloud. Performing authentication of smartphones in the cloud using behavioural authentication has been suggested by Chow et al. [9]. The security services proposed in [8] and [9] have been analyzed for feasibility by Stephanow and Tsvihun [10]. Chen and Itoh propose the concept of virtual smartphones in [14] which is mainly focussed on running the smartphone applications in the cloud. In their work the application itself resides in the cloud and only the user interface is made available to the smartphone user, thus the user virtually runs the application in the cloud. Paranoid Android [11], Security as Service-Reference architecture for Service Oriented Architecture (SOA) Security [12], and clone cloud [7] have been used in the architecture proposed for Security as a Service for smartphones in this thesis. The concepts proposed in [13][14][15] focus on offloading computation from the mobile device to the cloud.

To the knowledge of the author, this is the first work focused on identifying and structuring a *generic* architecture comprising possible security services that could be provided in a cloud for all smartphone platforms. However, the other works mentioned above have presented ideas for providing *individual* security functions as services in the cloud.

3 Smartphone Security

This chapter outlines the security aspects of a smartphone. The security objectives are discussed in section 3.1. The most prominent and frequent types of threats are presented in section 3.2 and the various infection channels through which these threats can possibly occur are summarised in section 3.3. Section 3.4 presents the various security functions that can be used to protect smartphones against some of the threats. The limitations of smartphones in providing these security services on the device itself are discussed in section 3.5.

3.1 Security Objectives

A well secured system should provide confidentiality, integrity, availability, and accountability. We will describe each of these security objectives in more detail below.

3.1.1 Confidentiality

Confidentiality refers to preserving the privacy or secrecy of information, i.e. preventing unauthorized disclosure. This requires that the information be kept in encrypted form and that only an authorized party can access this information in unencrypted form. In a smartphone, the confidentiality of information that is stored in the phone and that is transmitted from the phone should be ensured – this implies that (1) the information is kept in encrypted form or that the physical & logical device has to be protected and (2) that only encrypted information is transmitted (thus just before transmission is the *last* time that the information could be in an unencrypted form).

3.1.2 Integrity

Integrity refers to the protection of information from unauthorized, uncontrolled, or accidental alterations. Proper authentication, authorisation, and access control mechanisms can help protect the integrity of data. Maintaining information integrity refers to the protection of data from attacks and disaster. In the case of a smartphone, this requires integrity checks of the operating system and application software, ensuring the integrity of the data that is stored and transmitted over the network, and protecting the smartphone data in case of theft (this last implies that there is a copy of the data stored in a location other than in the phone – hence this other copy could be accessed if the physical phone is stolen). The copy of the data that is stored separately from the phone is often referred to as a backup copy of the data. In this way, it is possible to recover an unaltered version of the smartphone data. This data can be stored either in one location or spread in a redundant fashion across multiple servers. (See for example the cryptographically redundant storage of encryption keys in [46].)

3.1.3 Availability

The system needs to provide service preferably without interruptions, but in any case there should be rapid recovery after a service interruption. The importance of this objective depends on how crucial the service is to the on-going needs of the person or organisation that depends upon this service. In the case of network attached systems, the system should be resistant to Denial of Service (DoS) Attacks. DoS attacks could be made against the smartphone itself or against the service. Both types of DoS attacks should be avoided in order to ensure availability of service to the smartphone users.

3.1.4 Accountability

Accountability refers to the ability to account for the activities of an individual or an entity in the system. This can be implemented by utilizing logging and monitoring services within the system. These logs can be used to prevent individuals from denying their actions, thus achieving non-repudiation. Logs might also be important for understanding *retrospectively* what happened when a fault, error, fraud, or intrusion is discovered. Accountability could be very important for corporate smartphone usage, as a corporate smartphone user might be subjected to a variety of policies and regulations. The logs and monitoring services might be used to detect (after the fact) violations of one of these policies. Note that these corporate users might be subject to restrictions on what applications can be installed on their smartphones in order to ensure that some policies can not be violated (by only allowing applications that enforce the given policies). Additionally, these corporate users might even have restrictions on whom they can communicate with in order to enforce limitations on the spread of information or to prevent access to information that is not permitted.

3.2 Threats to Smartphones

In earlier times the probability of mobile phone threats were comparatively low when compared to PCs, as the devices generally were **not** programmable by anyone other than the vendor and the phones themselves were generally behind firewalls and network address translation devices operated by the network operator. Today smartphones offer PC like functionalities, hence are at risk of being attacked by similar threats to those encountered by PCs. Malware can be installed on the phone via Short Message Service (SMS) messages, Multimedia Messaging Service (MMS) messages, email, documents, web pages, etc.

The portability, convenience of usage and the functionalities of smartphones help their users to perform day-to-day activities such as sending e-mails, social networking, on-line banking, etc. – thus users will enter (and may also store) sensitive information on their devices. For many users their smartphone is the device that they use to access nearly all services, hence such a smartphone is a high value target for attackers. Note that these attacks can take the form of passive attacks based upon accessing sensitive information (for example, bank account & PIN number) or they can be active attacks (for example, causing the phone to send premium SMS messages or place calls to premium numbers operated by the attacker).

A number of different kinds of threats that affect smartphones are described in the following subsections. These do not represent all of the different kinds of attacks that can be made, but simply highlight some of the most common types of attacks.

3.2.1 Denial of Service (DoS) Attacks

According to a report from the European Network and Information Security Agency (ENISA), a smartphone could be used to launch distributed attacks such as Distributed Denial of Service Attacks (DDoS) against a target [16]. The report claims that the increasing popularity, complexity of smartphones, and their growing list of vulnerabilities will make this platform a valuable target for launching such attacks.

DoS attacks against the smartphone itself can flood the device, intentionally drain the battery, or consume other limited resources (such as memory, Central Processing Unit (CPU) cycles, port numbers, etc.). Flooding is possible by sending a large number of packets of data. Furthermore, these packets might be intentionally corrupted packets to cause the smartphone to request retransmissions – leading to attack amplification. Attacks that intentionally drain the battery attempt to keep the device active (see for example Buennemeyer, et al. [50] for a discussion of how to detect such attacks). Such an attack might mislead the user into believing that they have a defective battery or smartphone.

3.2.2 Malware

Mobile malware (i.e., malicious software) is software that can harm a mobile device without the owner's informed consent. This “harm” could be compromising the security objectives described above, causing economic damage to the user (for example, by running up their bill for a service), or actually physically damaging the smartphone itself. Malware includes viruses, worms, spyware, Trojan horses, and so on. The increased functional complexity of smartphones has increased the number of malware attacks.

A recent example of mobile malware is “Ikee.B”, the first iPhone worm created with distinct financial motivation [17]. It forwards any financial information that is stored on the iPhone to the attacker. Additionally, the worm scans for other vulnerable iPhones via the phone’s Wireless Fidelity (Wi-Fi) interface and third generation mobile telecommunications (3G) networks. The worm infects any vulnerable iPhones that it finds. Vulnerable iPhones are those that (1) have a Secure Shell (SSH) application installed to allow remote access to the device, but still have the root password configured as “alpine” (the factory default for this application) and (2) are jailbroken. A jailbroken iPhone refers a phone that has been hacked by the user in order to install third party software that has **not** been approved by Apple.

“ZeusMitmo.A” is a trojan targeted at phones running the Symbian OS [18]. If this Trojan is installed on the mobile phone, then all Short Message Service (SMS) messages can be monitored. Additionally, the attacker can utilize the application as a backdoor by sending commands via SMS. This Trojan was specifically designed for stealing SMS messages containing mobile Transaction Authentication Numbers (mTANs) [19]. This is ironic as mTANs were designed so that a bank’s customer did not have to have a physical list of transaction authentication numbers, which are used to authenticate e-banking transactions. However, it illustrates the general principle that crimes follow where the money is.

3.2.3 Social Engineering Attacks

Social engineering in a security context refers to trickery or deception for information gathering by manipulating people into performing actions or divulging sensitive information. Social engineering to spread malware has become commonplace in the Internet. An example is Commwarrior.C which utilizes information from a user’s address book and saved messages to create “believable” messages that a target is likely to read and reply to, enabling it to install itself on the target’s smartphone[47].

Phishing attacks are another type of social engineering attack that can occur on smartphones. Application stores (app-stores) for smartphones facilitate phishing attacks if the attacker can place fake applications on the site by disguising them as legitimate apps [20]. “09Droid” [21] for Android phones is an example of such an attack. Although different vendors utilize different validation and approval processes before mobile applications can be placed in their app-stores, due to the increasing numbers of new applications it is difficult to maintain a high level of confidence in the integrity of applications – irrespective of the vendor’s platform and their policy regarding new applications [22].

3.2.4 Theft

The small size, high value, and the incredible amount of valuable data that a smartphone may carry make smartphones increasingly attractive to thieves. Losing a smartphone can pose significant threats to the owner of the smartphone. Such a loss can also affect their employer if the smartphone is used for corporate or government use. (See for example, the news article which shows the increase in the number of laptops and mobile phones mislaid and the risks posed by them [54].)

3.3 Infection Channels

Smartphones can become infected through a wide range of infection routes. The following subsections detail each of the possible infection channels.

3.3.1 Bluetooth

Infection through Bluetooth™ depends on physical proximity of the attacker to the infected device. It requires the smartphone's Bluetooth connection to be switched on, sufficient signal strength, and that the phone is in its discoverable mode. Because there are no intermediaries between the infected device and a potential victim it is difficult to remotely monitor this infection route. Cabir is a well known Bluetooth worm that runs on the Symbian Series 60 platform and spreads among Bluetooth enabled devices that are in discoverable mode [23].

It might seem that the user is responsible for getting infected through their device's Bluetooth, as most of the settings for the Bluetooth connections are based on the preferences of the user. However, since most of these settings can also be affected by software – a clever attacker can infect a smartphone with a virus that will changes these settings, and then spreads itself via the phone's Bluetooth interface.

3.3.2 SMS / MMS

Malicious software can spread to mobile devices by attaching a copy of itself to an SMS/MMS that is sent from the infected mobile device. Commwarrior is an example of a worm that can spread through MMS [24]. The worm is capable of browsing the phone's phonebook and sending MMS messages to contacts in the phonebook, thus infecting these devices when the MMS is opened.

It should be noted that it is possible to send an SMS to the phone itself and to the SIM card in the phone – thus bypassing the user interface of the phone. There are over the air programming kits available to developers to develop SIM card applications, see for example Giesecke & Devrient's SmartTrust® Wib™ (a dynamic SIM toolkit interpreter), SkySIM® etc . There are even SIM card based browsers for feature phones, such as Giesecke & Devrient's StarSIM®.

3.3.3 Internet Connectivity

Most smartphones can be connected to the Internet using Wi-Fi, General Packet Radio System (GPRS), Enhanced Data rates for Global Systems for Mobile communication (GSM) Evolution (EDGE), or 3G networks. Smartphones run similar risks as fixed devices to become infected through viruses contained in downloaded files, cross site scripting, etc. (For examples, of some of these attacks against home routers see [48].) Skulls is a Trojan horse that masquerades as a useful application and convinces the user to download and install it from the internet through shareware sites [25].

3.3.4 Portable Memory

Usage of secure digital memory cards is commonplace in smartphones. Many smartphones such as the Samsung S8500 Wave [26] can support upto 32 GB SD memory cards[*]. Cardtrap is a trojan that affects Symbian smartphones by installing several Windows viruses, worms and trojans to the phone's Multimedia Card (MMC) [27]. These Windows applications are designed to be invoked by the user when they insert the memory card into a PC or when they transfer these programs to their PC (for example when synchronizing their smartphone and PC – see section 3.3.5). Note that the Trojan also attempts to disable applications in the smartphone itself.

3.3.5 Connection to other devices

Smartphones are often connected to fixed devices to copy information to the SD card or to synchronise data such as contact lists between devices. This kind of connection facilitates the transfer of the malware from the fixed device to the smartphone or the reverse. A crossover virus that is executed on the user’s Microsoft Windows PC can search for handheld devices that are connected to the PC in order to infect them [28].

3.4 Security Functions

This section describes a number of security functions that are used to protect smartphone data and applications.

[*] Today there are upto 2Terabyte SD memory cards, but smartphones generally only support much smaller capacity cards.

3.4.1 Encryption

Encryption is the process of transforming plain text data to a cipher text data using an algorithm and a key. Many different encryption algorithms can be used to protect the data. The user's security requirements determine the strength of the algorithm and key length that are chosen. Since valuable information is being stored in smartphones and/or transmitted via the network, this data should be encrypted to ensure that the confidentiality of the information is not compromised. Only the authenticated entities that possess the key for decryption will be able to easily retrieve the data. Advanced Encryption Standard (AES) and Triple Data Encryption Standard (3DES) are two of the encryption algorithms that are widely used in smartphones [31]. Few smartphones provide encryption for the data that is stored on the device, however this situation is changing with BlackBerry providing encryption for data stored on media cards, Microsoft introduced its Encrypted File System (EFS), Nokia's Wallet application, iPhone 3GS's hardware encryption and LUKS for Android phones. Encrypting this stored data could prove valuable if the smartphone is lost or stolen.

3.4.2 Digital Signatures

Digital signatures verify the authenticity of the sender of the message involved in a communication. Validating this signature gives a reason for the receiver to believe that the message was sent by the claimed sender. Therefore, digital signatures can also be used to provide non-repudiation.

Digital signatures can also be used to verify message integrity, as any changes to the message after it has been digitally signed will invalidate the signature. Hence, smartphones should support digitally signed messages and validate all incoming messages.

Digital signatures can also be computed over code, thus it is possible to sign applications – both when stored in a file system and when in memory. Today many PCs designed for enterprise use contain a Trusted Platform Module (TPM) that can be used together with features in the processor and Basic Input/Output System (BIOS) to check that the intended software is what is actually running. Research has been done to develop a Mobile Trusted Module (MTM), see for example [49].

3.4.3 Anti-virus

Anti-virus software for a smartphone can be used to detect, prevent, and remove malware from the phone. It can be used with malware such as viruses, worms, trojan horses, spyware, etc. Signature scanning is a common method of malware scanning. This method is based upon searching for known patterns of malware in the executable code and files. To identify zero-day attacks, heuristics can be used that can detect slight variants of malicious code. Some anti-virus providers have concentrated on providing software for smartphones, for example F-Secure and Kaspersky [29][30]. However, new malware targeting smartphones will be released, as this platform is both vulnerable and valuable for attackers.

3.4.4 Anti-Theft

As described in section 3.2.4, smartphones are at risk of being stolen, thus necessitating anti-theft security functions to protect the data in the phone from mis-use. One method is to reduce the risk is to remotely wipe the data from the smartphone once it is lost. Another method is to remotely lockout out access to the data. Kaspersky Mobile Security has an anti-theft feature where the phone can be remotely blocked by sending a pre-defined SMS or the user can opt to wipe the data, and it is also possible to track the location of the mobile device using the Global Positioning System (GPS) feature by sending an SMS with a passcode [30]. It should be noted that remotely wiping out the data or remotely locking out access both carry a risk that an attacker could trigger the remote operation to perform a DoS attack. Turning on tracking of the phone remotely can also be a threat to personal integrity – when the phone is being used by its owner.

3.4.5 Authentication

Authentication mechanisms can be used to ensure that only the authorized user is able to access the functions of the device. However, passwords do not offer robust authentication as most users choose weak passwords due to the keyboard constraints of smartphones, because users do not want to have to remember many passwords, and because most users do not value security (except when they have been negatively affected by a loss of privacy, integrity, etc.). Password authentication can be enhanced by forcing a user to choose a strong password; however, one can not expect better results than for attempts to force the user of strong passwords on desktop machines. When the security requirements are high, more advanced and strong authentication mechanisms such as two factor authentication, behaviour based authentication, voice recognition, key stroke based authentication could be used [31][32][33].A smartphone can also exploit speaker recognition as described in [61] for authenticating the user of the smartphone.

3.5 Limitations of smartphones

In this section, some of the limitations of smartphones with respect to the implementation of security functions are highlighted. As the security requirements of smartphones are considered to be high, appropriate security functions should be used. Although smartphones are starting to face threats similar to that of PCs and laptops, a comparison of the technical specification of today's laptop computer and a smartphone show significant differences in their capabilities. Due to the resource constraints, methods that should be used to provide security on smartphones often need to be adapted or alternative means selected.

Smartphones have traditionally had more limited computational capacity and storage, while the users have different expectations of the operating time of smartphones than they do for their laptop. The security functions that are essential to protect smartphones are usually resource intensive. Shin et al. [34] presented a study of the performance of the Secure Socket Layer (SSL) on a PDA to quantify resource use on a handheld device. In their study each step of the protocol was identified and compared to that of the same protocol running on a laptop. Their measurements show that the CPU clock speed is not the only factor that affects performance. In particular how the PDA accesses the network and the associated latency of this access also contribute to degraded performance. Cryptographic operations consume more energy on the PDA than on the laptop, even though the laptop is computing results much faster than the PDA (in fact the laptop takes only 31% of the overall execution time of the PDA). Rifa-Pous et al. [35] show that energy consumption of a PDA with an ARM processor while performing cryptographic operations is 16% higher when the battery contains 25% of its fully charged power than when the battery is fully charged; this appears to be because the processor takes the same amount of time to perform the calculations, but as the battery voltage is lower when the battery has less charge remaining - the current has to be higher in order for the processor to run at the same speed, hence the total power consumption is higher.

An experiment to characterize the energy profile of various Internet Protocol (IP) services in smartphones has been carried out by Zayas and Gomez [36] in order to identify the energy consuming aspect of these services and to improve them. Battery performance is one of the main criteria considered in their work. A detailed analysis of power consumption in a smartphone has been performed by Carroll and Heiser [37] to quantify the major factors influencing power consumption in different use cases for a smartphone. Their results show how different energy consumption profiles impact the battery life of the smartphone.

While the regular operations of a smartphone (as discussed in the above studies) can have significant impact on energy consumption, resource intensive security functions such as an anti-virus scan through all the data in the smartphone and real-time virus scanning (running in the background) can deplete battery resources rapidly. This implies that either such an approach to virus checking should **not** be done or that this operation should be offloaded to a resource rich computing environment.

4 Existing Architectures

In this chapter, some of the key architectures whose concepts have been used in deriving the architecture proposed in chapter 5 will be described. A brief description of the architecture of Opera Mini, BlackBerry, Paranoid Android, Security as a Service for SECTISSIMO framework, Clone cloud, and smartphone mirroring architecture are discussed in sections 4.1, 4.2, 4.3, 4.4, 4.5, and 4.6 respectively.

4.1 Opera Mini

The Opera Mini architecture has been discussed in [38] and [39]. Opera Mini is a mobile web browser designed specifically for smartphones and PDAs. Opera Mini is derived from the Opera web browser used on personal computers [38]. Opera press releases [51], [50], and [52] showcase a tremendous increase in the number of people using this high performance browser. Figure 3 shows the architecture of Opera Mini which is very similar to the earlier Wireless Application Protocol (WAP) model. The mobile phone only needs to support Java in order to run the Opera Mini client. This architecture does not directly provide security services for the mobile phone, but it illustrates a simple solution that can optimize the performance of mobile web browsers at the cost of most of the security objectives that we described earlier!

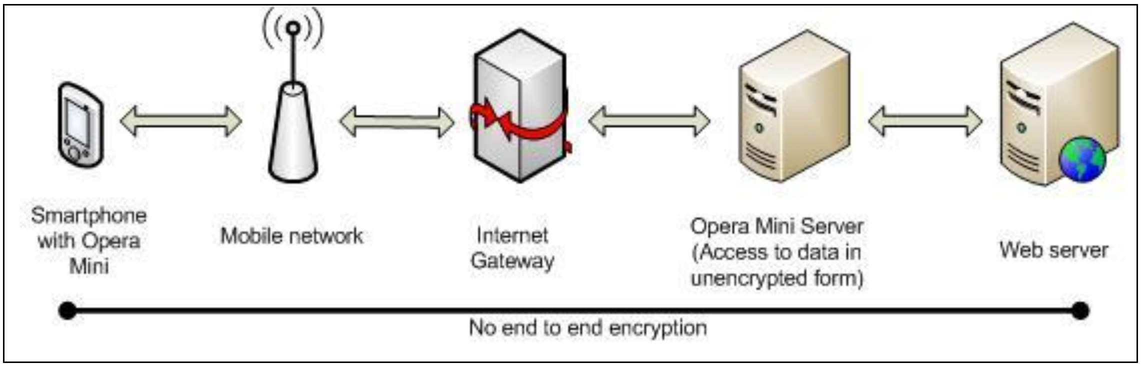

Smartphone with Opera Mini
Mobile network
Internet Gateway
Opera Mini Server (Access to data in unencrypted form)
Web server
No end to end encryption

Figure 3: Opera Mini Architecture adapted from [39]

As shown in Figure 3, requests from the Opera Mini web browser pass through Opera's Mini servers which forward this request to the actual web server and processes the response (for example, compressing the response) before sending a response to the mobile device. This can improve the speed and considerably reduce the amount of data transferred to the mobile phone, thereby enabling a good web browsing experience even with constrained resources.

The Opera Mini browser in the mobile phone fetches the contents of the website through the Opera Mini server which acts as a proxy/transcoder server that can translate Hypertext Markup Language (HTML) with Cascading Style Sheets (CSS) into a more compact format. It can also resample & compress the images to suit the screen of the smartphone. In order to perform these transformations the Opera mini server needs to have access to the unencrypted webpage. Therefore, end-to-end encryption is not compatible with this method, for example making this unsuitable for highly sensitive financial transactions **unless** the user trusts the Opera Mini server software. It should be noted that this is the same sort of security model that WAP adopted, thus requiring either trusted intermediates to operate these Opera Mini servers or each website needs to operate its own Opera Mini server.

Considering the security aspects of the Opera mini architecture it does **not** offer protection against social engineering attacks, malware, DoS, and theft of the smartphone. As it is an architecture concerned with the performance of mobile web browsers, it does not provide security against other infection channels. Confidentiality is compromised at the transcoder server as the information needs to be available to the transcoder in an unencrypted form. The transcoder server modifies the data from the web server before it is provided to the mobile device thereby improving performance at the cost of integrity of the service. The availability and accountability security objectives are outside the scope of this architecture. A 256 bit Rivest Cipher 4 (RC4) is used for encryption, a 1280 bit RSA[†] asymmetric encryption is used for key exchange, and the Secure Hash Algorithm (SHA)-256 is used for hashing. A password manager allows storing or clearing of passwords from the history of accessed browser pages. This architecture does not influence the phone's physical security in any respect. Bandwidth consumption may be considerably reduced using the Opera mini server, which can in turn reduce energy consumption of the smartphone. Opera also offers Opera Mobile browsers which are capable of providing end-to-end encryption at the cost of performance, such browsers should be used for highly sensitive transactions, for example, financial transactions.

4.2 BlackBerry Enterprise Architecture

The BlackBerry® Enterprise Architecture is an integrated solution from the Research in Motion (RIM) Group. This architecture (shown in Figure 4) consists of the BlackBerry Enterprise Server (BES) and the other components of the "BlackBerry Infrastructure". The BlackBerry Enterprise Architecture is considered a robust architecture in terms of security [42].

[†] RSA is the security division of EMC (http://www.rsa.com/node.aspx?id=1002)

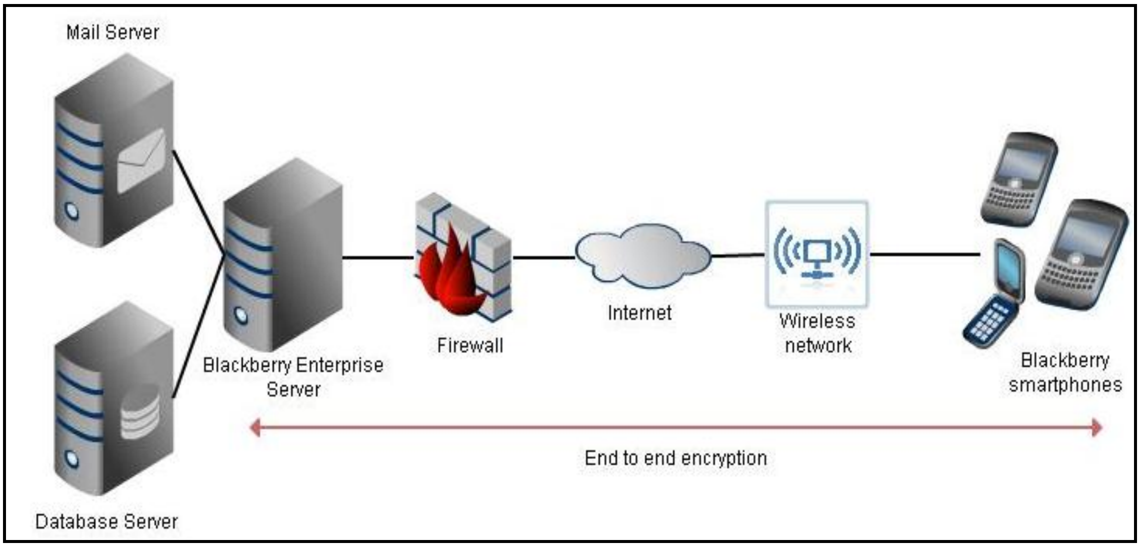


Figure 4 : BlackBerry Enterprise Architecture adapted from [45]

BES and it components provide the following functionalities [43]:

- Providing tools and data from an organisation's applications to the end users equipped with BlackBerry Smartphones
- Monitors other BES components
- Processing, routing, compressing, and encrypting the data
- Communication over the wired and/or wireless network utilizing the BlackBerry infrastructure

RIM ensures the following in terms of security:

- On-device security; and
- Secure connections between the BES and the BlackBerry Infrastructure (specifically the BlackBerry or BlackBerry-enabled devices).

When the BlackBerry device is turned on by the user, the processor in the device executes instructions from an internal Read Only Memory (ROM) which is the root of trust for the BlackBerry devices. This internal ROM reads the bootROM code which is stored in flash memory and verifies the signature of the bootROM using the public key stored by the processor. The processor continues to run only if this verification is successful allowing secure boot. End-to-end security is provided for all data that is transmitted between a BlackBerry Smartphone and the BES. The BlackBerry Enterprise Architecture provides support for Public Key Infrastructure (PKI). Additionally, both Secure Multipurpose Internet Mail Extension (S/MIME) and Pretty Good Privacy (PGP) are supported for email. Two factor authentication is also available for BlackBerry Smartphones using the RSA Secure ID [44].

The BlackBerry Enterprise solution uses symmetric key cryptography to encrypt messages and data that are sent over the transport layer. Confidentiality is ensured by allowing only intended message recipients to view the message content. Integrity is provided by the use of one or more message keys to protect every message that is sent from the BlackBerry smartphone. The use of encryption and message keys means that third parties cannot modify or decrypt any messages. The master encryption key is known only to the BES and the BlackBerry device. Both the BES and BlackBerry will reject any incoming message which is not encrypted with the correct master encryption key. The availability of the various services provided by RIM depends on the contract with the service provider and the options enabled in the smartphone by the user. RIM offers a large number of Information Technology (IT) policies which can be set according to user or organisational preferences in order to avoid misuse of the smartphone. The BlackBerry administration service also controls and monitors all the BlackBerry devices using over-the-air commands and policies thereby providing accountability.

The BlackBerry Enterprise solution offers effective protection against malware due to the use of application control policies, IT policies, and code signing. Theft is handled by remotely issuing administrative commands that can lock the device in order to protect the data or simply erase all the data.

The BlackBerry Enterprise solution offers services that can protect the smartphone from all the infection channels presented in section 3.3, but use of these services is at the discretion of the user. Bluetooth connections require pairing with the device. Additionally, it is possible to encrypt all of the data that is sent over the Bluetooth link. IT policy rules are used to offer SMS/MMS security by controlling the messages and unsecured messaging can be disabled. BlackBerry devices can connect to their desktop managers through a secure communication channel and a secret password is exchanged between them. BlackBerry Enterprise solution encrypts data sent over Transmission Control Protocol/Internet Protocol (TCP/IP) and the BES encrypts data between specific components of the BlackBerry infrastructure. BlackBerry content protection offers protection to the data stored on the device and the external file system encryption level IT policy rule in the device settings on the BlackBerry influences the security used for external memory devices.

Although the BlackBerry Enterprise Architecture is considered robust, provides a well constrained execution environment, and secure end-to-end communication infrastructure, it is a closed system hindering further exploration and/or exploitation. Furthermore, the security functions are limited to the BES and the BlackBerry infrastructure components. When the BlackBerry device communicates with another device outside the BlackBerry infrastructure, most of the security features may be unavailable. This architecture can serve as an example of a security as a service architecture for smartphones.

4.3 Paranoid Android

Portokalidis et al. proposed Paranoid Android in [11]. Paranoid Android offers versatile protection for smartphones. This architecture views security as just another service at a higher level that can be hosted in the cloud. In this section we will present details of this security model for Android devices.

Figure 5 illustrates a high level overview of the Paranoid Android architecture. The basic idea is to run a synchronised replica of the smartphone in a security server in a cloud. Since the cloud server has abundant resources, intensive security checks are carried out on the clone in the security server in the cloud. A record and replay technique is used for the replication, thus a tracer application in the phone records all the information that is required to accurately replay the target application's execution. This trace information is transferred over an encrypted channel to the security server where the replica of the Android phone runs in an emulator. The replayer replays the execution in the replica based upon the trace data. Security checks are performed on the replica. A proxy server in the fixed network is used by the Android phone to connect to the internet so that this proxy can provide a copy of all inbound traffic to the replica. The replayer accesses the proxy server to retrieve the inbound traffic that it needs for the replay. The amount of data that the tracer needs to send is minimised (with respect to its network traffic) as it only needs to provide information about the state changes and the local user and device input and output to the Android device – since the network traffic can be replicated by the proxy and the replica has a copy of all of the Android device's starting state.

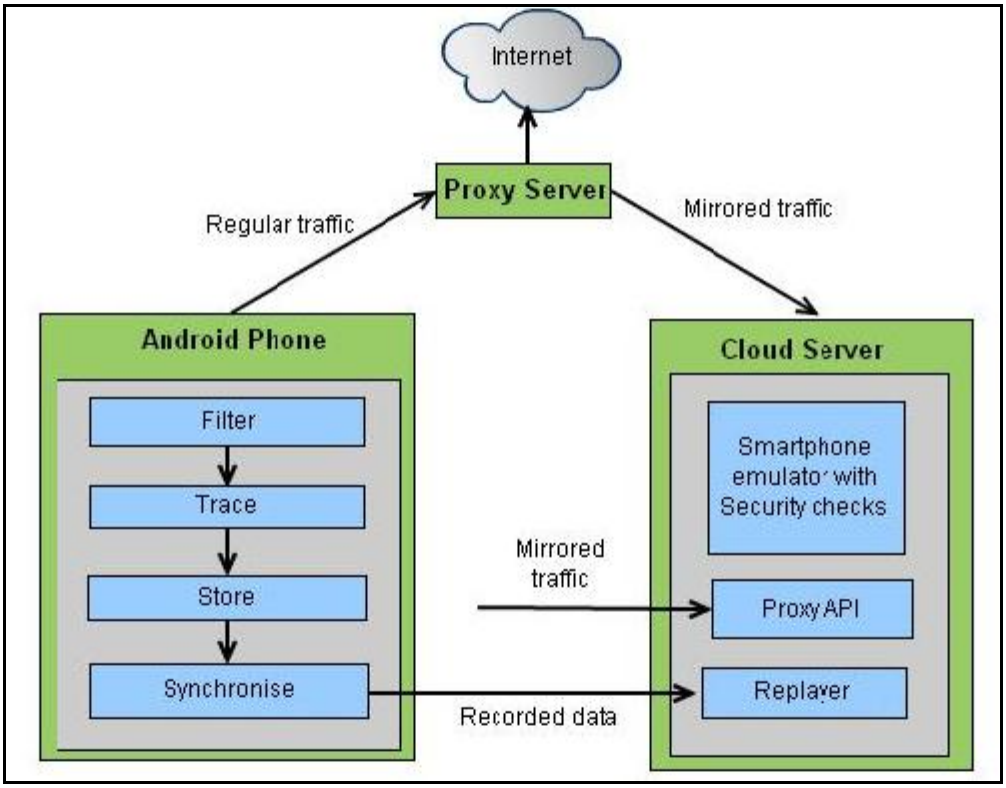


Figure 5: Paranoid Android Architecture adapted from figure 1 of [11]

More specifically only the non-deterministic inputs and events that influence the execution flow need to be captured as part of the trace. In practice this means that all the data that is transferred from user space to the kernel and all the data that is transferred from the kernel to user space via system calls are recorded by the tracer. Loose synchronisation is adopted to synchronise the smartphone and the replica in the cloud in order to conserve the battery resources of the phone. Virus scanning, dynamic taint analysis, and memory scanners are some of the security measures that could be implemented in the server. Additional techniques that can be employed include user behaviour modelling and speculative computing (i.e., the replica could take advantage of its resources to compute ahead of the Android device and provide the phone with responses to system calls that could be available as soon as the system call is made).

The Paranoid Android architecture focuses more on attack **detection**. The prototype implementation works on the Android platform and is specific to the Android architecture. Zero day attacks and memory resident attacks that have targeted mobile phones can be handled. These attacks gain importance because the Java Run Time Environment (JRE) version optimised for mobile phones is not as secure as the native Java virtual machine. This architecture can be extended to protect against social engineering attacks, theft, or DoS by implementing suitable security functions in the cloud; but it is not designed to prevent their occurrence. The proxy server is not sufficiently protected, as the incoming and outgoing traffic is stored in the proxy which can be subjected to threats such as cache poisoning.

The confidentiality objective does not seem to be fulfilled, as there is no mention of keeping the data encrypted in the smartphone or in transit. Integrity of data requires trust of both the proxy and the cloud server. Also man-in-the-middle attacks are possible as there is no use of secure connections. The integrity of the trace is preserved by use of a Hash-based Message Authentication Code (HMAC) function. This architecture assumes loose synchronisation, as it is assumed that the smartphones cannot always be connected to the internet. Therefore availability is affected, as the security functions in the cloud are not always available to the smartphones. A software bug or a failed attack on the execution trace manifests as synchronisation errors and the smartphone is restored back to a clean state which implies that the actual cause for the failure is difficult to be identified.

The security functions in the cloud try to detect attacks caused through all the infection channels. Security functions such as anti-virus, dynamic taint analysis, memory scanners, and system call anomaly detection could be used, although the use of authentication, digital signatures, and authentication have not been discussed. In this architecture, they propose having secure storage only for the execution trace as it can destroy the functioning of the system if the phone is compromised and the synchronisation procedure disabled.

An evaluation of the prototype implementation of Paranoid Android shows that the transmission overhead can be kept well below 2.5 Kbps during periods of high activity, such as during browsing and audio playback. It also shows that the battery life is reduced by 30% due to the user space implementation of the tracer, but the authors suggest that implementing the tracer in the kernel can significantly improve the battery life - although this has not (yet) been proved.

4.4 Security as a Service (SaaS) for SECTISSIMO Framework

Memon et al.[12] propose the SaaS paradigm as applied to the SECTISSIMO framework. SECTISSIMO is a platform independent framework for security modelling and implementation [40]. Since enforcement of all the security functions at the service endpoint can cause a significant burden at the service end point, SECTISSIMO shifts the security burden from the service endpoint to shared and dedicated security services running within a security domain. The result is a hybrid combination of integrated and decoupled security.

Integrating the entire necessary security infrastructure such as policy repositories, protocol execution engines, monitoring agents, and so on would be an expensive solution at the service endpoint. Instead the security requirements of the service endpoint focus on security decisions such as token validation and authorisation according to a security protocol. Once a security decision has been made, it needs to be enforced at the service endpoint. Therefore, isolation of the execution of the security protocol and the communication of the security decision to the endpoint is achieved by implementing them as separate services. Only the enforcement of security decisions is carried out at the service endpoint.

Figure 6 shows the SECTISSIMO architecture based on the SaaS approach. The upper portion of the figure shows a health care system with various service endpoints. The global request and response handlers (shown on the left in the upper half of the figure) are integrated with the service endpoints that intercept the incoming and outgoing messages to provide security at the service endpoint. The security proxy handler within the service endpoint implements the basic tasks such as encryption/decryption, signing/signature validation, security decision enforcement, and key exchange.

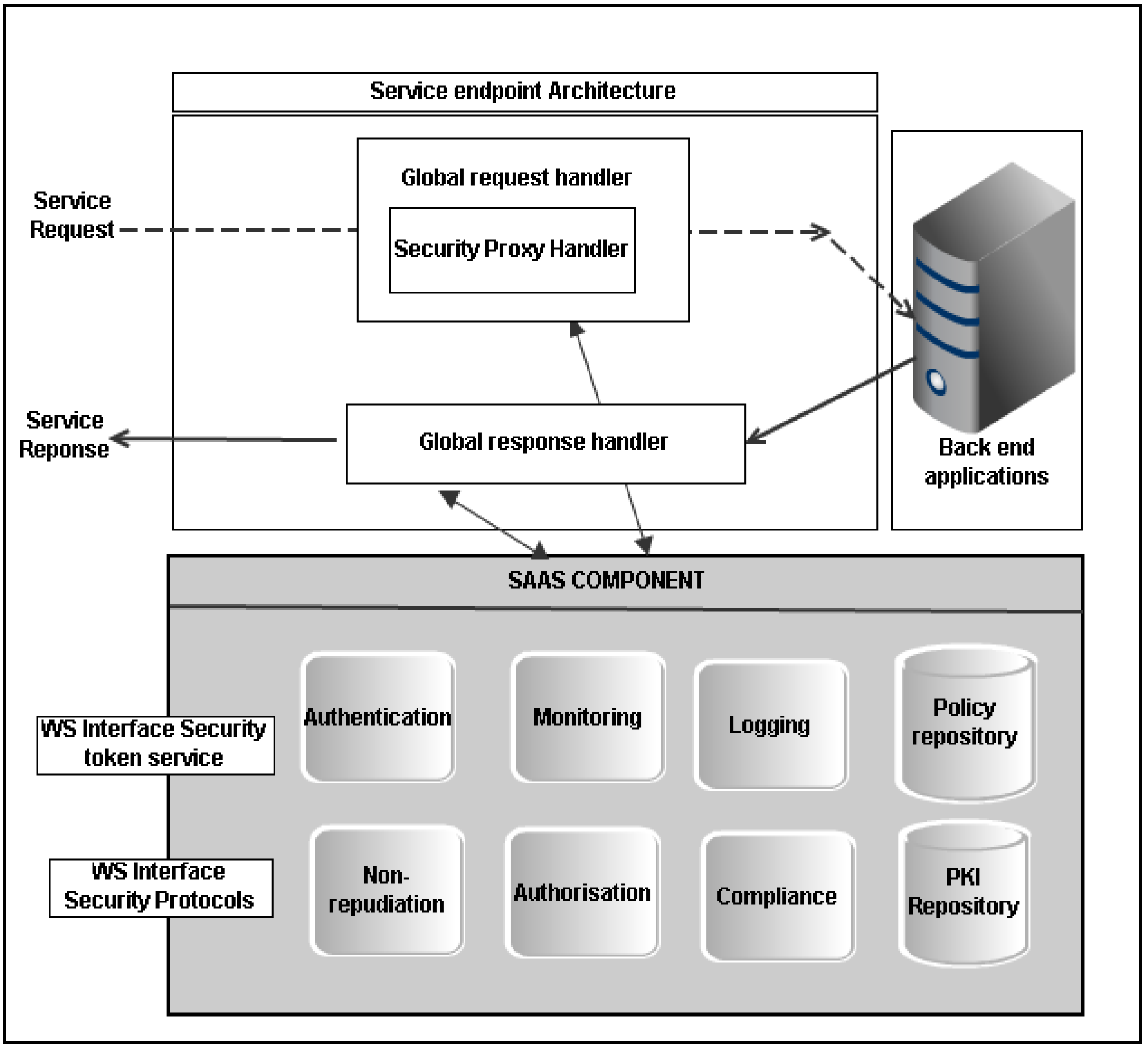


Figure 6: SaaS approach for SECTISSIMO adapted from figure 2 of [12]

The SaaS component (shown in the lower half of the figure) is deployed by the security domain to provide shared security services such as authentication, authorisation, non-repudiation, logging, monitoring, and compliance to the service endpoints of a particular security domain. The policy repository and the PKI repository are in turn used by these security services. The Web Service (WS) Interface Security Token Service and WS Interface Security protocols are used to communicate with other security domains to resolve the identity of the endpoints of other domains and for the execution of security protocols with the endpoints of other domains.

Being a generic architecture for security services, all the security objectives described in section 3.1 have been taken into account. It provides effective isolation between the various security functions that are carried out in the central SaaS component and the endpoint. The architecture allows addition of security functions to the SaaS as required by the target system based on the sensitivity of applications handled by the system. A system that adopts this architecture can modify the security function components as required for the specific infection channels or attacks that it needs to address.

4.5 Clone Cloud Architecture

Distribution of computation between the smartphones and the cloud resources in the form of a clone cloud architecture has been suggested by Chun and Maniatis [7]. The concept behind the clone cloud architecture is to seamlessly offload execution from the smartphone to a computing infrastructure. Resource intensive processes or portions of processes are performed by the smartphone clone in the cloud. These results are then merged with the state of the smartphone which resumes execution. The clone can also be used as a backup if the smartphone is lost (as the clone has a copy of all of the data that the smartphone has upto the last synchronization point).

This process of splitting the computation between the smartphone and its clone is referred to as “augmentation”. This augmentation of computation is being done in four steps:

1. Creation of a clone of the smartphone in the cloud;
2. Periodic or on-demand synchronisation of the smartphone and the clone;
3. Applications can be augmented (whole applications or augmented pieces of an application) automatically or upon request; and
4. Results obtained from the augmented segment of computation from the clone are synchronised with the smartphone.

Figure 7 shows the clone execution architecture for the smartphone which aims to transform a single machine’s execution into a distributed execution. The smartphone and the clone have a replicator which is responsible for synchronising the clone state with the smartphone (on-demand or periodic) and a controller component on the smartphone that invokes augmented execution in the clone and integrates the results of the computation by the clone with the smartphone. The augmenter in the clone is responsible for execution in the clone and it sends its results to the controller in the smartphone for integration with the local application’s sate.

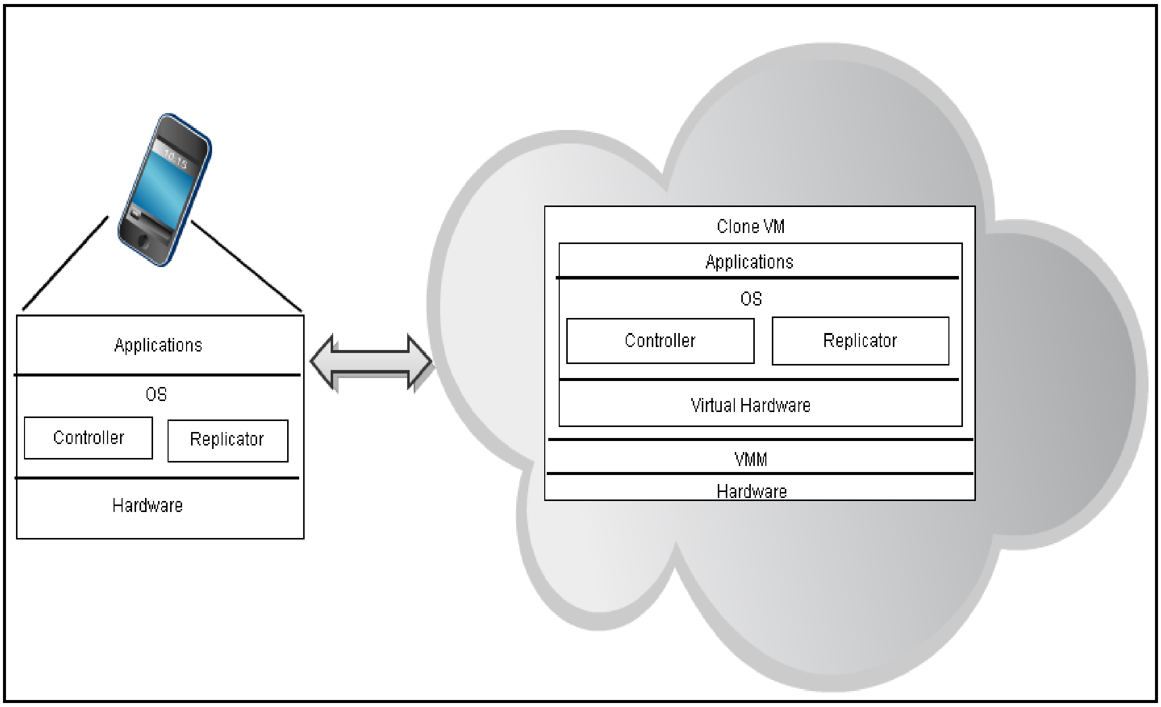


Figure 7: Clone Execution Architecture adapted from figure 3 of [7]

The Clone Cloud architecture does not focus on providing security for the smartphones, but presents an effective method for computation offloading from the mobile devices. Anti-virus for smartphones has been suggested as one of the functions that can use the resources of the cloud.

A prototype implementation of the Clone Cloud is demonstrated in [53]. In the Clone Cloud prototype, an application is partitioned by the use of a static analyser, dynamic profiler, and an optimization solver. The execution (migration and re-integration) points are defined where the application migrates part of its execution to the cloud. This takes place at a thread level and when the compute intensive thread completes execution in the cloud; its results are merged into the state of the smartphone. The other threads can still continue to run on the smartphone. Migration decisions are made based on criteria such as current network characteristics, CPU speed, and energy consumption at the smartphone.

The prototype implementation evaluation proves upto 20x speedup and 20x energy reduction. This architecture is more application oriented and focuses on seamlessly offloading some resource intensive operations off the mobile device. Executing security functions completely in the cloud is outside the scope of this architecture. Furthermore, the details of how a smartphone and its clone are kept synchronised have not been discussed. Therefore, analysis of how a clone would be created and managed needs to be considered if this architecture is to be adopted.

4.6 Smartphone Mirroring architecture

Zhao et al.[55] propose a framework to keep the mirrors of smartphones on a computing infrastructure in a telecommunications network thereby offloading heavy computations to the mirror as shown in Figure 8. The mirror server in the telecommunications network is capable of hosting a large number of virtual machines. Synchronisation between the smartphone and the mirror is achieved by replaying all the inputs to the smartphone in the same order at the mirror. Their framework is different from the others in the following two major aspects:

- The cloud is located in the telecommunication service provider’s infrastructure
- A smartphone is connected to its mirror through a 3G network

Modifications to the existing telecommunications network is necessary to enable the forwarding of all the incoming traffic of the smartphone to the mirror server. In this system design, it is considered that the changes in the state of the smartphone are triggered by user or network inputs which produce deterministic outputs. Thus they suggest that synchronisation can be achieved by replaying these inputs in the mirror in the order of their occurrence. Data caching applications and anti-virus scanning are the two services for smartphones that have been identified to benefit from this framework. However, this framework has *not* been implemented yet.

The mirroring smartphone approach does not concentrate on providing security services to the mobile phones, although it seems to have a similar concept as proposed in the next chapter (chapter 5). Replaying of smartphone inputs can be considered as one important aspect to achieve synchronisation in the proposed architecture in chapter 5.

Furthermore, this approach does not consider Wi-Fi and Bluetooth connections and caters only to the internet connections through 3G networks.

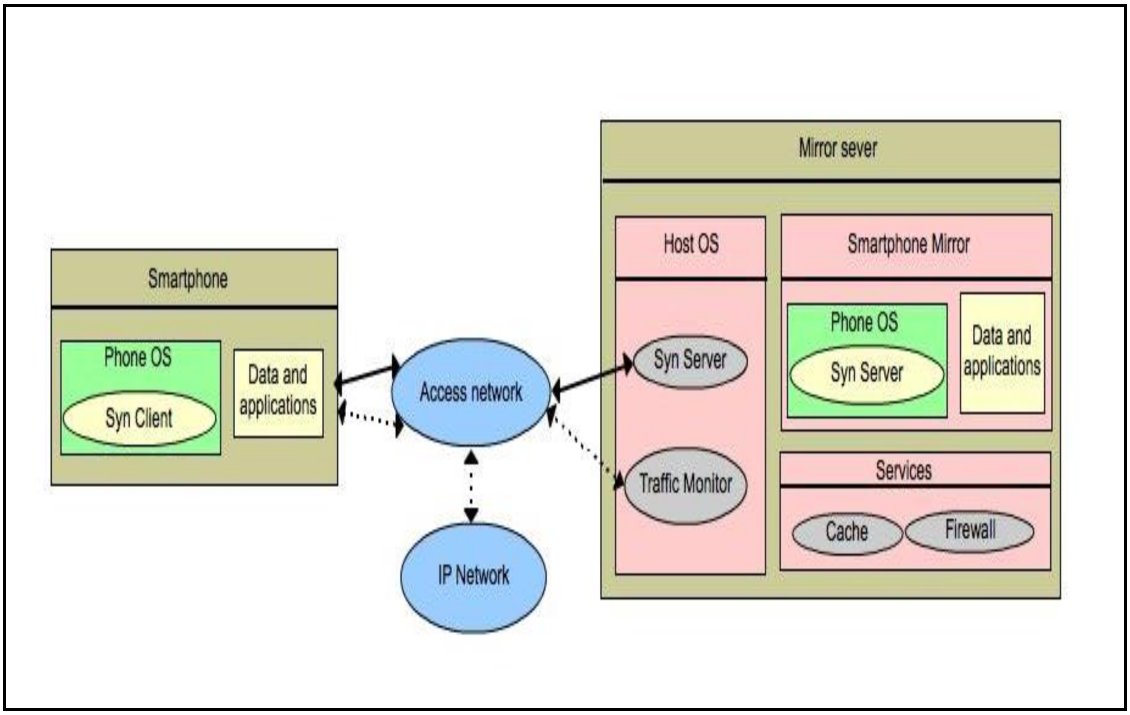


Figure 8: Smartphone mirroring architecture adapted from figure 1 of [55]

5 Security as a Service Architecture for Smartphones

The security as a service (SeaaS) architecture is a generic architecture for all smartphone platforms that primarily encompasses the provision of security services in the cloud for smartphones. The term SeaaS has been adopted from the famous “As a service” notion used for the cloud services such as IaaS, SaaS etc. Although the architecture can be extended and used for general purposes, it has been designed to be used by business users in a corporate scenario. Section 5.1 provides an overview of this architecture. Sections 5.2 to 5.7 detail the various components of the architecture. Section 5.8 presents the security functions that can be provided as cloud services to the smartphones. Section 5.9 deals with message sequences for some sample usage scenarios of the smartphone when connected to the Internet.

5.1 SeaaS Architecture

The basic concept behind the architecture for Security as a Service (SeaaS) framework for smartphones is shown in Figure 9 where some of the security services are offloaded to the cloud environment. It portrays smartphones at one end and the cloud server offering the security services to the smartphone on the other end.

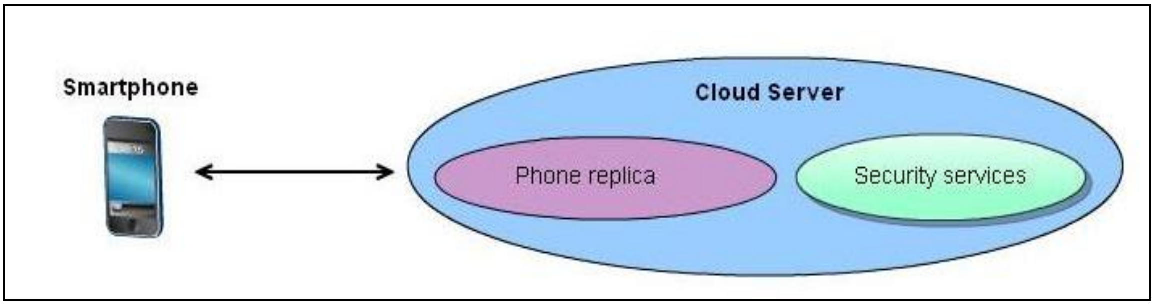


Figure 9: Basic concept of SeaaS

The architectural framework depicted in Figure 10 is an extension of the basic concept depicted in Figure 9 and encompasses smartphones as the end device, a server farm in the cloud which includes a proxy server which controls the traffic in and out of the mobile device, and additional servers to realize cloud services that provide security services to smartphones.

Multiple virtual machines for the replicas could be run in the cloud server as and when needed. The security functions can be deployed either in the replica Virtual Machine (VM) or in the native OS of the cloud server based on the kind of functionality it provides to the smartphone. Each virtual machine uses hardware virtualisation on top of the physical hardware of the cloud server.

The Sync Module in the phone and the replica VM are responsible for synchronizing the states of the smartphone and the replica. This synchronization can either occur at fixed time intervals or be an on-demand synchronization approved at the discretion of the controller in the smartphone. This controller can make its decision based on available bandwidth, estimation of possible energy consumption, and estimated energy consumption before the smartphone will be recharged. Furthermore, the controller can also be used to decide which security functions are most appropriate to offload depending on the network conditions in which the smartphone is most likely to operate and limitations of the specific smartphone based on its technical specifications. The service manager in the replica VM is responsible for the execution of the security functions and sending the results to the interpreter in the smartphone. A cloud based proxy is used to provide anonymity service, intercept the incoming and outgoing traffic to be acted upon by the security functions, providing a caching service, and implementing a firewall service.

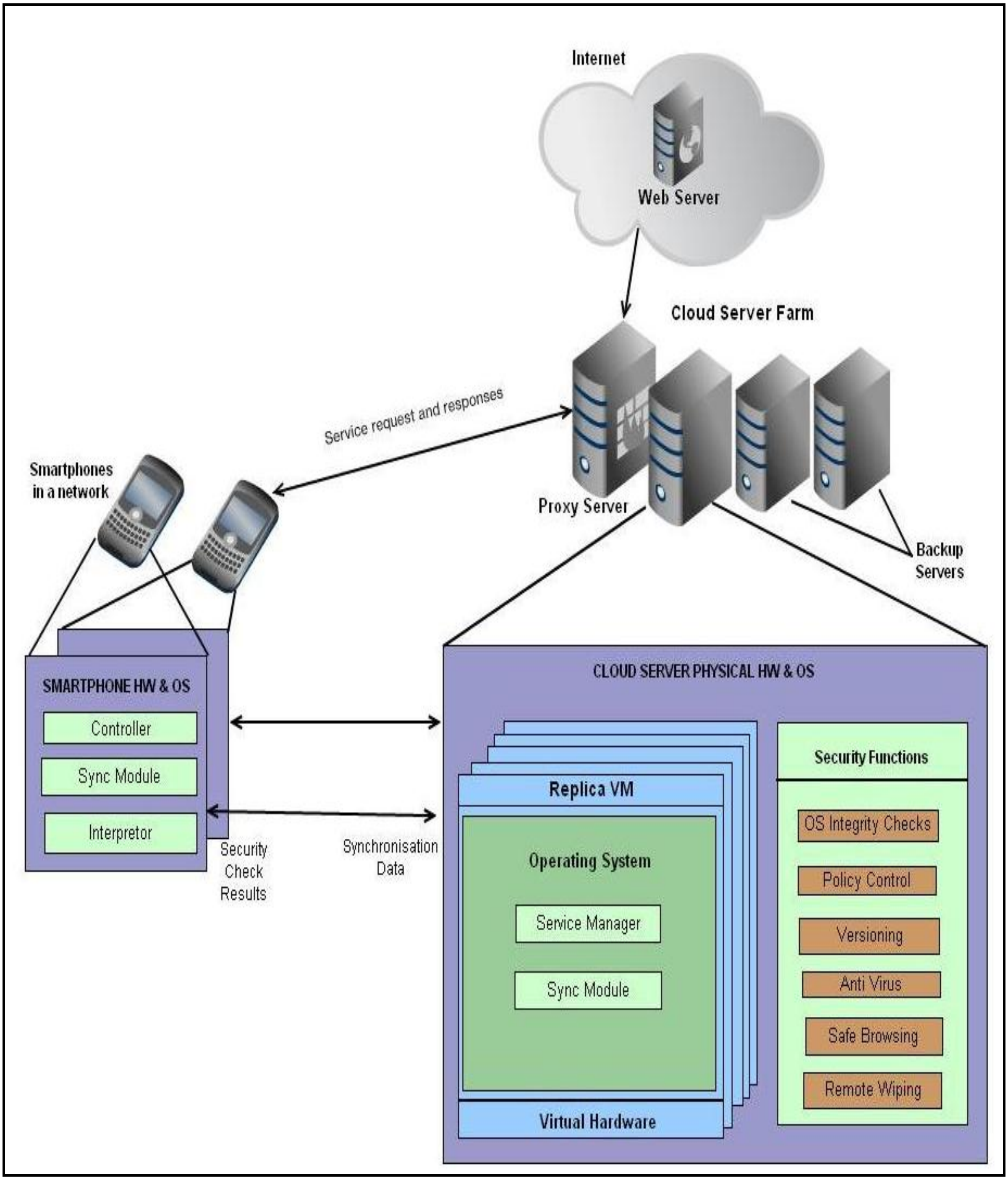


Figure 10: SeaaS Architecture for Smartphones

A replica of a smartphone can be thought of as a copy maintained in the same state as the state of the smartphone. This means that the copy contains the smartphone OS files and files accepted by the user to be synced to the replica. There are many ways for creating and maintaining the replica in the cloud. In the following subsections, the individual components of the architecture are discussed.

The components of the proposed architecture include a sync module, an interpreter, and the controller in the smartphone; sync module and service manager in the replica and security functions in the cloud server; the cloud based proxy server and the backup servers. Figure 11 shows the components of the smartphone and the cloud in the SeaaS architecture.

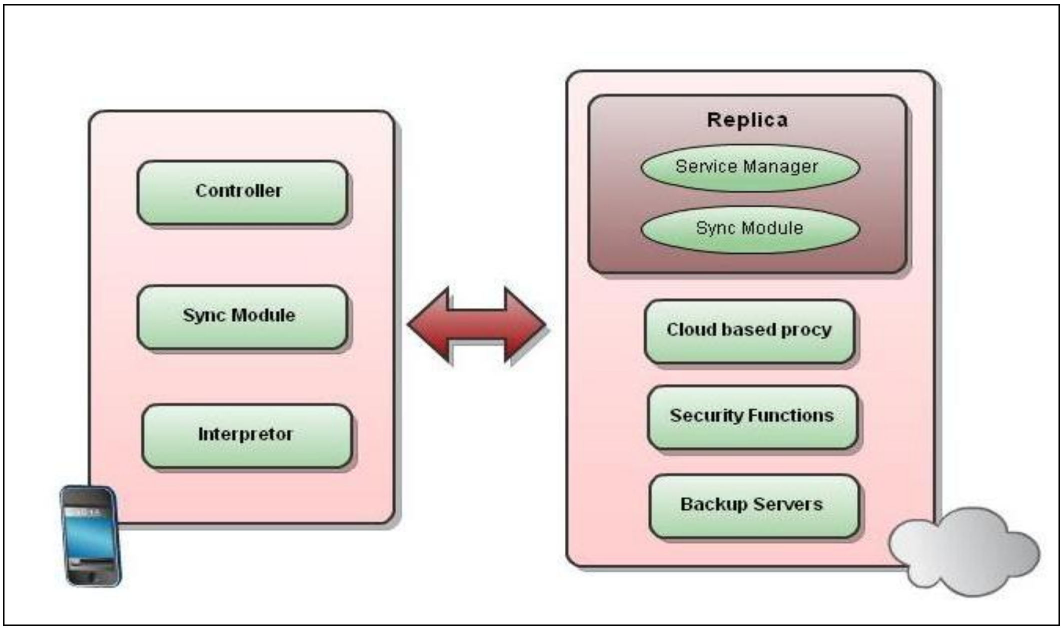


Figure 11: Components of the smartphone and Cloud

The following set of assumptions has been made in this architecture:

- All kinds of smartphones are taken into consideration, ranging from lower to higher end models. Battery resources are scarce in every smartphone even though higher end models may offer better CPU performance.
- This architecture is mainly targeted for deployment in a corporate setting and the cloud infrastructure can be *in house* in the case of very high security requirements.
- Cloud servers are capable of hosting any number of replicas as required.

- The cloud server farm is trusted by the smartphone users as all their information is also stored in the cloud.
- Smartphones are connected to the cloud through a high throughput, low delay internet connection (preferably Wi-Fi or 3G link).
- The targeted infection channel is the internet; although the architecture can be extended and used to monitor other infection channels.
- All the network connections involved are secure *end-to-end*.

5.2 Sync Module

The sync module in the phone and the replica are responsible for keeping the states of the smartphone and replica synchronised. A smartphone user is allowed to choose the files which (s)he wants to synchronise to the cloud server by the use of the sync folder in the smartphone. In addition to these files, the smartphone operating system files are also synchronised. In general, we assume a synchronisation of a replica and the smartphone is complete at time ‘t’, if at time ‘t’ the replica in the cloud has an identical copy of the smartphone operating system files and the files which the user accepted to synchronise. In case of high security requirements, strict synchronisation i.e frequent synchronisation should be enabled. In other cases, loose synchronisation (less frequent) could be adopted. Achieving the desired synchronisation assumes sufficient network connectivity such that the smartphone is able to update its state to the replica in the cloud.

The Sync Module can be categorised as follows:

Data Sync Data Sync establishes a similarity in the smartphone and replica’s state and data when it is requested to do so.

Time Sync Time Sync decides the frequency of updates to the replica to keep the smartphone and replica synchronised and notifies the data sync unit to start the synchronisation process. This can be at regular intervals (i.e., periodically), on-demand, or both. Any deviation from the expected behaviour of the time sync unit can be attributed to the commands from the controller based on various factors like bandwidth availability, battery consumption, etc.

Supposing the replica and the smartphone data were synchronised at time ‘t’, then the next synchronisation occurs at time ‘t+x’, x being a random or a fixed time interval. This will update the replica with the changes that have occurred at the smartphone during the interval ‘x’, provided the user approves to sync them. If there is stable and sufficient bandwidth, then synchronisation at fixed time intervals can be adopted. In this case, the time intervals basically determine the level of synchronisation. Additionally, on-demand synchronisation is also possible where the states of the smartphone and the replica are synchronised when it is desired. For loose synchronisation, the ideal time to synchronise can be identified by the use of a controller (as discussed in the next section).

The sync module is instructed by the controller as to what data needs to be sent to the replica in the cloud. The sync module in the replica is responsible for placing the data in the virtual machine’s state and file system appropriately.

5.3 Controller

The controller component is present in the smartphone and is responsible for tracking the data that needs to be sent to the replica to achieve synchronisation. Determining the kind of data that needs to be sent depends on how synchronisation is achieved.

Replication can be achieved in several ways. From the Paranoid Android architecture described in section 4.3, it can be seen that one way that replication can be achieved is by using a record and replay technique, where all the system calls from kernel to user space are tracked for a safe replay at the emulator in the cloud. Alternatively, in the smartphone mirroring architecture presented in section 4.6, the user inputs to the smartphone are tracked to effectively replay the actions at the smartphone mirror in the same order as occurred at the smartphone.

In the SeaaS architecture, tracking user inputs to the smartphone could be a set of data that can be used as any change to the state of the smartphone is stimulated by some user action on it, or via the network, or via some Input/Output (I/O) device or sensor on the smartphone. The network inputs are already made available at the replica through the proxy server. To maintain the consistency in the file system of the smartphone and the replica, any new file that has been added to the file system of the smartphone (from sources other than the internet in case of continuously connected operation) is also sent to the replica. For example, a user may receive a file through Bluetooth. Although the user action of accepting the file input is tracked and reported to the replica, the contents of the file received via Bluetooth is not made available at the replica. Therefore such a new file and other I/O & sensor input will have to be transferred to the replica by the controller provided the user accepts to do so by enabling these options in the settings interface in the smartphone.

The controller is also responsible for making the phone-replica synchronisation decisions as required. As mentioned earlier in section 5.1, the controller can make its decision based on available bandwidth, estimation of possible energy consumption, and estimated energy consumption before the smartphone will be recharged.

5.4 Interpreter

The interpreter in the smartphone receives the results from the service manager in the cloud and interprets them. Based on the result obtained, it acts on them according to an established policy and appropriately imitates the user or waits for a user action.

5.5 Service Manager

A service manager in the replica VM controls the execution of the various security functions that are present in the cloud server and provided as a cloud service to smartphones. It assumes responsibility to invoke the security functions and sends results to the interpreter in the smartphone.

5.6 Cloud based proxy

A cloud based proxy is appropriate for the architecture as it can scale well and support an enterprise's set of smartphones by offering them a pool of services. The proxy server is deployed in the cloud server farm.

The cloud based proxy can act as a firewall to the enterprise's network. To do so, the network traffic in and out of the smartphone must go through the proxy server. In this way, it is possible to apply security checks to the incoming traffic *before* forwarding this traffic to the smartphone, thereby making it possible to achieve attack prevention. This is possible but it incurs the cost of additional delay – and the assumption is that this added delay will be acceptable to ensure security.

A safe browsing service can be offered to the smartphones by scanning the requested web pages for any malicious behaviour and the user can be notified accordingly.

Anti-virus scanning can be applied to the incoming traffic, thereby protecting the smartphone from getting infected.

As discussed in the Opera mini architecture in section 4.1, the cloud based proxy can also be used to enhance the performance of smartphones by delivering web page contents to the mobile phone in a compact manner adapted for the specific smartphone's screen size. In this case the proxy server needs to have access to the contents of the data that is delivered to the smartphone to optimise it making it suitable for the screen size.

The requested web pages could be cached by the proxy server for quick retrieval by the smartphones when accessed in the future. The cached requests and responses can reduce the bandwidth consumption of the smartphones. Furthermore, the replica hosted in the cloud server can query the proxy server to obtain the required data during synchronisation. The proxy server will also provide an anonymity service to the smartphone hiding the identity of the smartphone in the enterprise network.

5.7 Backup Servers

Backup servers in the cloud can be used to host the smartphone replicas in case of failure of the main server, thereby enabling a replacement phone to provide the same services as the original phone or another replica to provide the same services as another replica with minimal disruption beyond the delay to initialize the required processes.

5.8 Security functions

The proposed architecture lists some of the many security functions that can be provided as a cloud service to smartphones. The security functions can be placed in the native OS of the cloud server or in the emulated replica. The choice of location depends on the kind of security function provided by the service. The security functions to be implemented in the initial prototype are:

- Anti-virus,
- OS integrity checks,
- Policy control,
- Browser protection,
- Versioning and Remote Wiping, and
- Secure Storage.

5.9 Message Sequences for sample scenarios

In this section examples of message sequences are shown for two scenarios: web page access and user download of a file.

5.9.1 *User accessing a web page*

The proxy server in the cloud intercepts the incoming traffic from the web server and then forwards the response to the smartphone after completing the security processing (See Figure 12). In this sequential approach known malware can be identified and prevented from being transmitted to the smartphone; however, it comes at the cost of increased delay in delivering content to the smartphone.

For a system that cannot tolerate the delay due to security processing, the incoming traffic could be sent to the smartphone and then the security checks could be done in parallel. In this parallel approach, outbound traffic from the smartphone could be blocked until the smartphone has received security evaluation results from the cloud. If no malicious content is reported, then the outbound traffic could be allowed – otherwise this output should be blocked and immediate action taken to revert the smartphone to a safe state (See Figure 13).

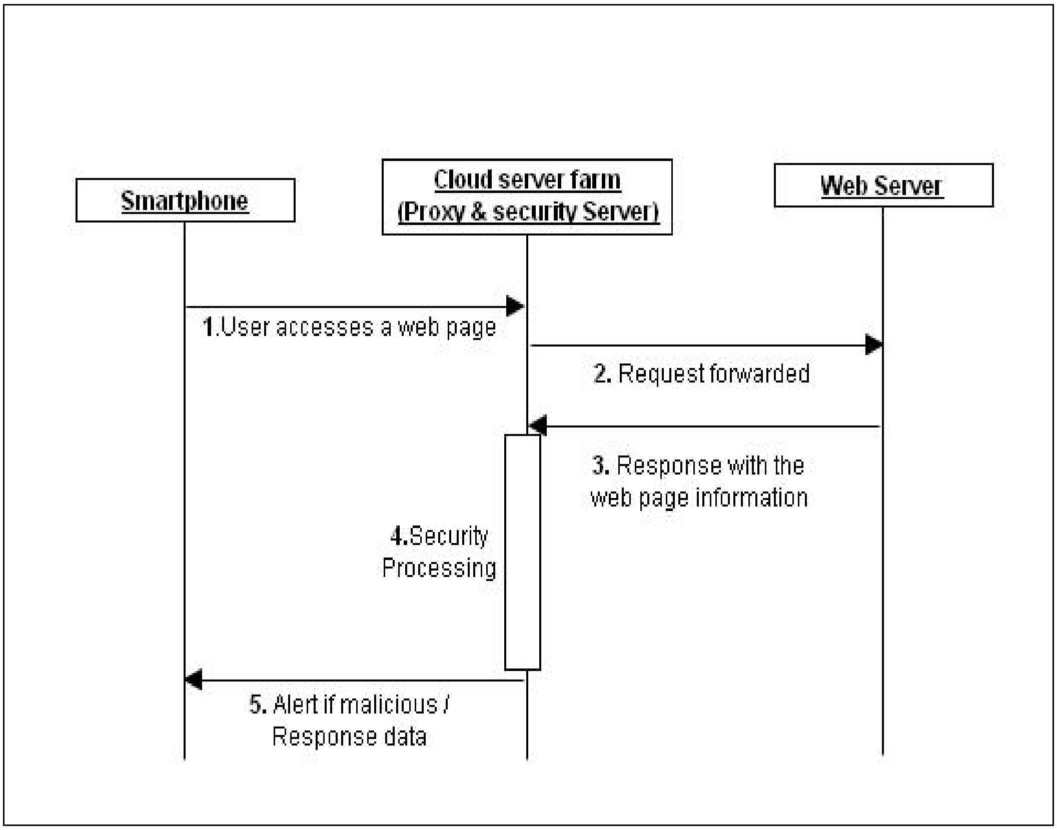


Figure 12: Attack prevention in a sequential approach

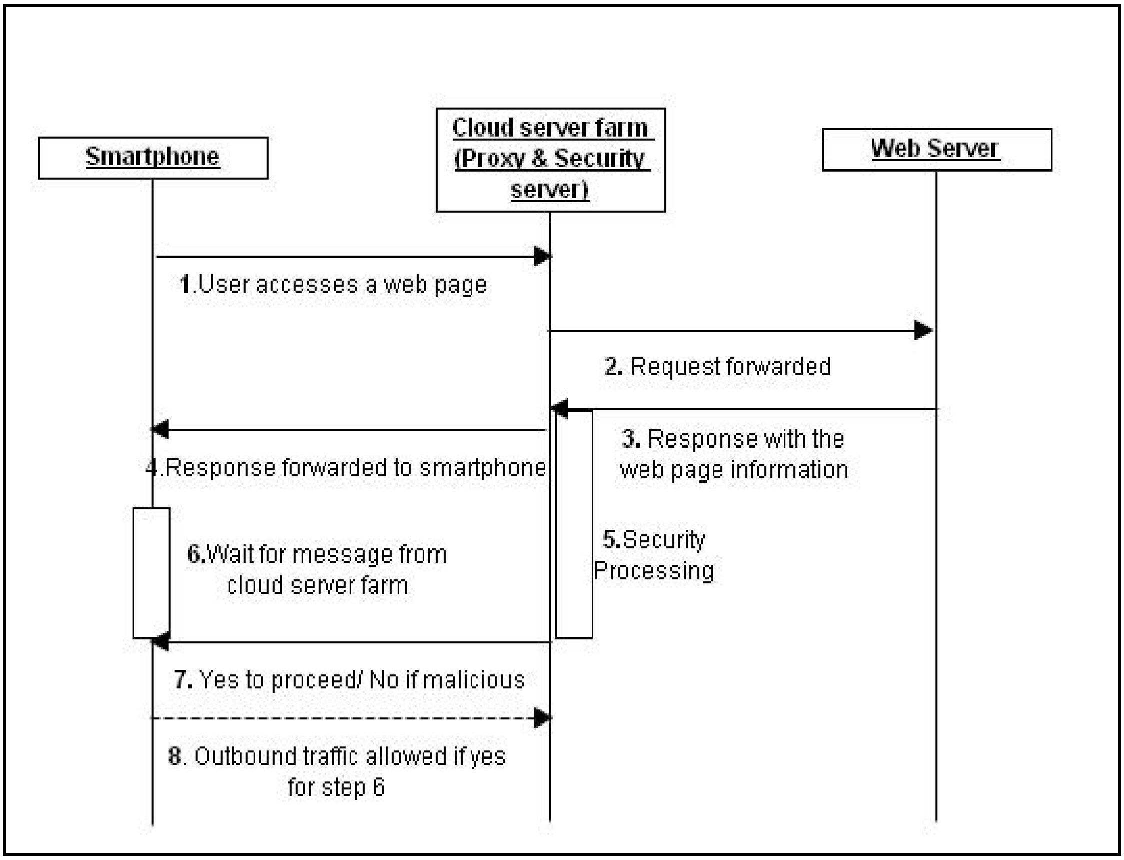

Smartphone
Cloud server farm (Proxy & Security server)
Web Server
1.User accesses a web page
2. Request forwarded
3. Response with the web page information
4.Response forwarded to smartphone
5.Security Processing
6.Wait for message from cloud server farm
7. Yes to proceed/ No if malicious
8. Outbound traffic allowed if yes for step 6

Figure 13: Attack detection and preventing further damage in parallel approach

5.9.2 User downloading a file

In this scenario a user wishes to download a file. The same two possibilities as explained in section 5.9.1 for the web page access scenario can be applied to this scenario. In the sequential approach, the file is downloaded from the web server to the proxy after which security processing is done on it. The file is then downloaded to the smartphone if the content is not malicious. In the parallel approach, the download is initiated to the smartphone at the same time the download occurs to the proxy. The cloud server can do the desired security processing and send the results to the smartphone, which if the file is not malicious would enable the user to use the file with lower delay than in the sequential approach. Figure 14 and Figure 15 show the sequential and parallel approaches respectively.

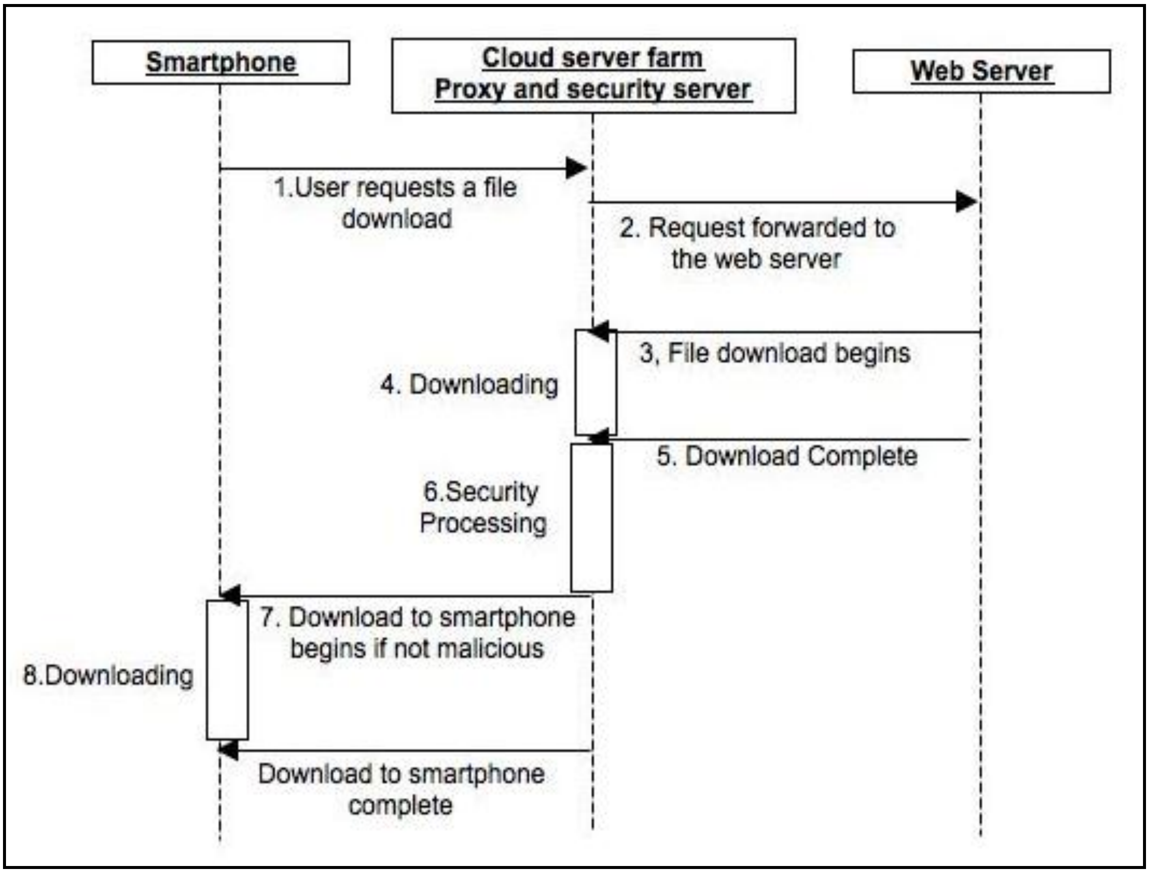

Smartphone
Cloud server farm
Proxy and security server
Web Server
1.User requests a file download
2. Request forwarded to the web server
3, File download begins
4. Downloading
5. Download Complete
6.Security Processing
7. Download to smartphone begins if not malicious
8.Downloading
Download to smartphone complete

Figure 14: File download in a sequential approach

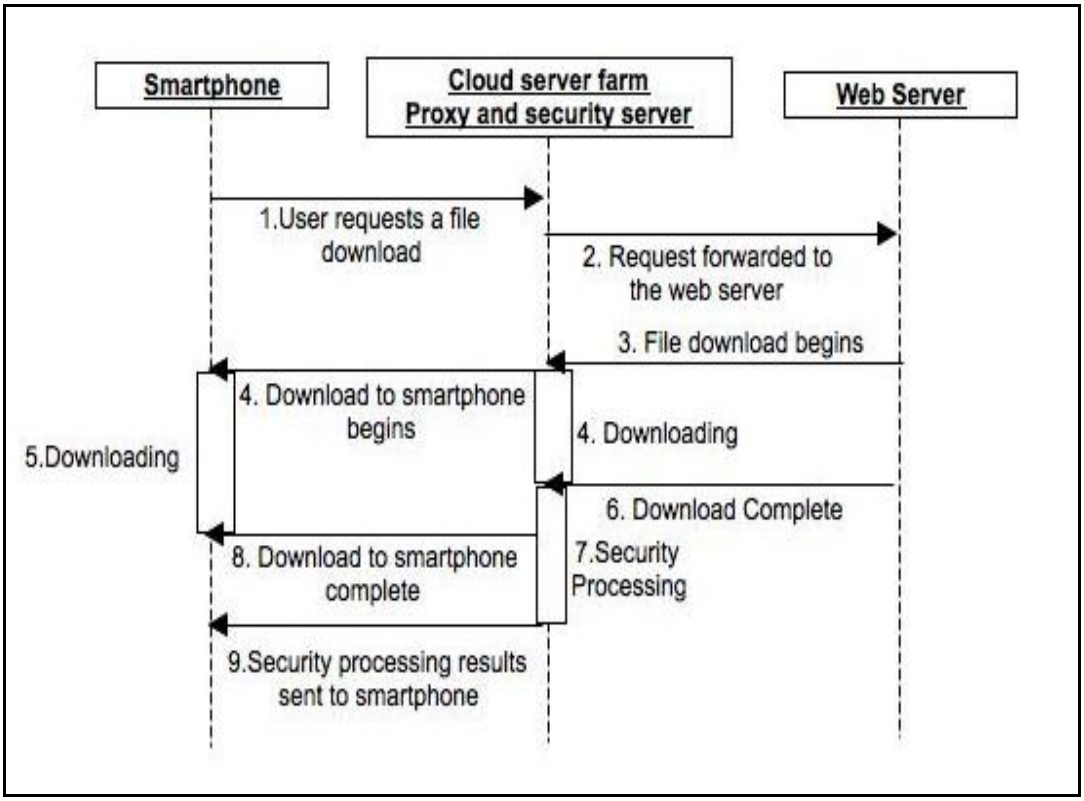

Smartphone
Cloud server farm
Proxy and security server
Web Server
1.User requests a file download
2. Request forwarded to the web server
3. File download begins
4. Download to smartphone begins
4. Downloading
5.Downloading
6. Download Complete
7.Security Processing
8. Download to smartphone complete
9.Security processing results sent to smartphone

Figure 15: File download in a parallel approach

5.10 Use Case Scenario for the SeaaS Architecture

The SeaaS architecture can be deployed in a corporate organisation set up. It is often seen that an organisation provides its employees with a laptop to be used for official purposes and to connect to the corporate network from home if necessary. This situation might soon change allowing the employer to provide smartphones instead of laptops to its employees. Then the user connects to the corporate network from his/her smartphone instead of a laptop.

Secure storage services provided by the cloud server can be used to access confidential information like non-disclosure agreement documents, project documents etc. Access control policies and monitoring services in the cloud can be used to log the activities of the specific user for future references if necessary. Policy control service can be used to enforce the policies for the employee using the smartphone. Safe browsing and anti-virus scanning services ensure real time protection to the smartphones preventing them from getting infected through the Internet connection.

6 Measurements

This chapter describes the measurements that have been made in the course of this thesis project. The chapter begins by examining how anti-virus software could be used in the proposed architecture. This is followed by some measurements of resource consumption, especially battery power consumption.

6.1 Anti-virus Performance Measurement: SmartPhone versus Emulation of a SmartPhone

We begin by describing some initial performance measurements of anti-virus scanning of the state of an Android handset. Amazon's Web Services was chosen to simulate a cloud environment. The initial instance configuration used for the virtual machine in the cloud is shown in the following table.

Table 1: Initial Amazon Web Service virtual machine configuration

613 MB memory
Up to 2 EC2 Compute Units (for short periodic bursts)
EBS storage only
32-bit or 64-bit platform
I/O Performance: Low
API name: t1.micro

However, this environment was not sufficient to support the installation of the Android Software Development Kit (SDK), thus the Amazon Web Services medium instance configuration was used (see Table 2).

Table 2: Amazon Web Services medium instance configuration

1.7 GB of memory
5 EC2 Compute Units (2 virtual cores with 2.5 EC2 Compute Units each)
350 GB of instance storage
32-bit platform
I/O Performance: Moderate
API name: c1.medium

The operating system running on this instance is Ubuntu Linux and an Android version 2.1 smartphone was emulated. Android Version 2.1 was chosen due to the availability of a physical Android 2.1 version smartphone (for the configuration of this physical phone see

Table 3), for comparison with the emulated version. The emulator and the smartphone contained the same information to facilitate the comparison.

Table 3: Specification of the physical smartphone used

HTC wildfire with Android 2.1	
CPU Processing Speed	528 MHz
Storage	ROM: 512 MB RAM: 384 MB
Battery type	Rechargeable Lithium-ion battery with 1300 mAh capacity

6.2 AVG anti-virus

The AVG anti-virus [58] software was used to scan the applications, settings, content, and the media files of the physical smartphone and the emulated smartphone. The smartphone was configured with 100 MB of content and media files, in addition to the default Android version 2.1 smartphone files.

6.2.1 Emulator

Figure 16 shows the CPU activity of the emulator during the anti-virus scanning process. The x-axis depicts past 60 seconds in history i.e it decreases from 60 sec to current time corresponding to 0 sec. The graph shows the activity of the 2 CPUs of the instance and it can be seen that signature scanning is quite resource intensive, as it consumes at its peak approximately 90% and 58% of the 2 CPUs cycles. Due to the use of 2 processors the scanning is quicker than when using a single CPU. The total time taken is about 7 seconds for the scan. Except for the gnome display process, all other processes of the instance were **inactive** when this measurement was performed.

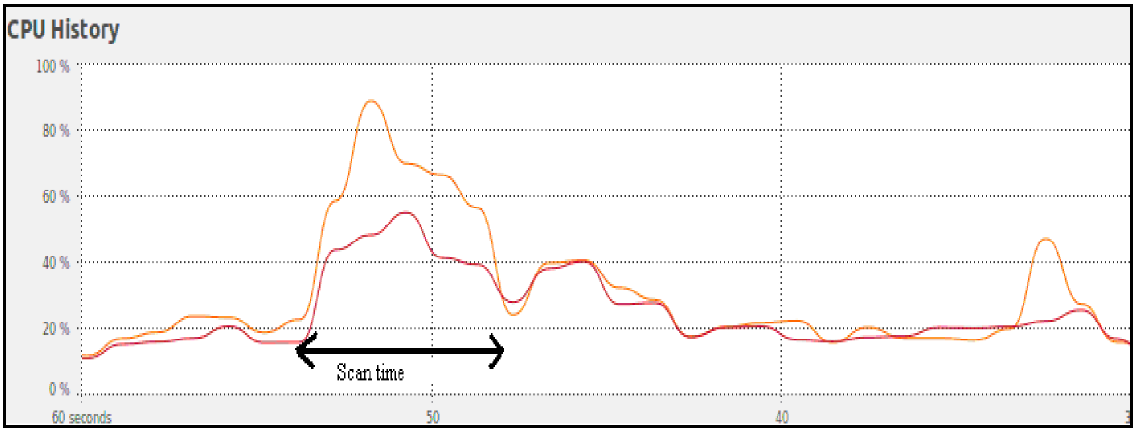


Figure 16: CPU activity as a function of time

The AVG anti-virus scanning running on the physical smartphone uses upto 85% of the CPU during its operation. The storage consumption is shown in Table 4. It includes the resident memory, shared memory and effective memory. Resident memory refers to the physical RAM installed in the smartphone. Shared memory refers to the physical or virtual memory that is shared between one or more processes. Effective memory refers to the amount of memory that gets freed when the application is exited. For a single scan of the device 1% of the battery's energy capacity was consumed. The total time taken for the scan was about 50 seconds.

Table 4: Memory consumption during scan with AVG anti-virus

Resident memory	29.6 MB
Shared memory	14 MB
Effective Memory	17.3 MB

6.3 NetQin Mobile anti-virus

To see if the performance of anti-virus scanning differs when using different anti-virus software another anti-virus application was used. In this case the NetQin Mobile anti-virus [60] software. As before, the software was run on both the emulator and an actual smartphone.

6.3.1 Emulator

Figure 17 shows the activity of the 2 CPU's of the Amazon instance while running the emulator during an anti-virus scan using the NetQin Mobile Anti-virus software. The signature scanning process took about 3 seconds to complete and consumed up to a maximum of 62% and 44% of the two CPUs. Except for the gnome display process, all other processes of the instance were **inactive** when this measurement was performed

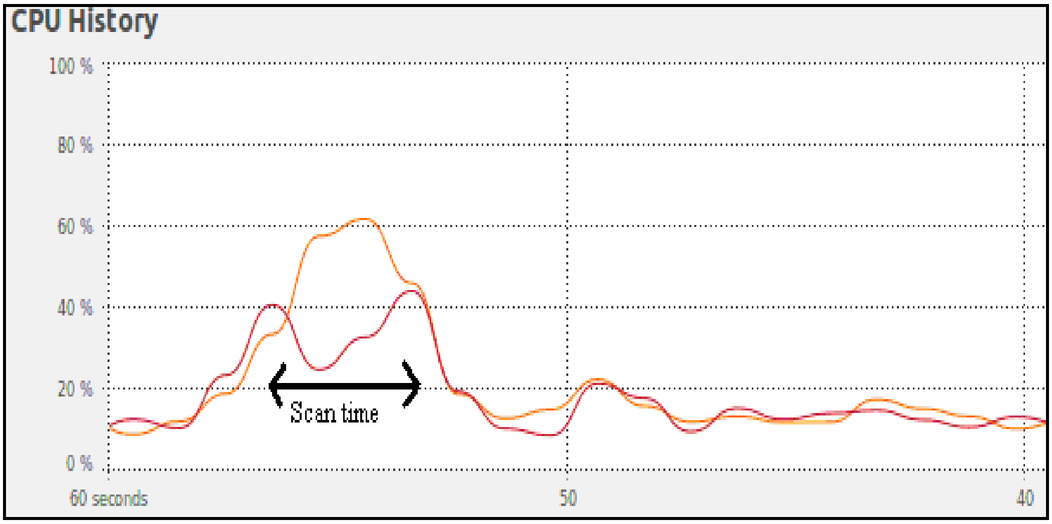


Figure 17: NetQin Mobile Anti-Virus running on the emulated smartphone

6.3.2 Smartphone

The NetQin mobile anti-virus scanning uses upto 71% of the physical smartphone's CPU resources. The amount of storage used while scanning is shown in Table 5. For a single scan of the device 1% of the battery's energy capacity was consumed. The total time taken for the scan was about 15 seconds.

Table 5: Memory consumption during scan with NetQin Anti-virus

Resident memory	28 MB
Shared memory	12 MB
Effective memory	16 MB

These preliminary measurements of anti-virus scanning on a smartphone and an emulation of a smartphone show that the time required to perform a given scan is much less on the emulated smartphone (due to the faster processors and use of dual processors) (see Table 6). In addition, scanning in the emulator did **not** require any battery consumption (beyond the energy required to synchronize the emulated image with the physical smartphone. Furthermore, the emulator in the cloud instance allows installation of multiple vendor's anti-virus software which could provide improved detection coverage.

Table 6: Summary of scanning times

Software	Physical Smartphone (seconds)	Emulated Smartphone (seconds)
AVG anti-virus	50	7
NetQin Mobile Anti-virus software	19	3

The virus scanning operation can also be performed in the native OS of the cloud server, for example, the Ubuntu Linux used in the above experiments. A sample scan was done with Clam anti-virus 0 in the Ubuntu Linux in cloud. The total number of signatures in the anti-virus was 953725 and the total number of scanned files was 2247. This scan took 6.11 minutes to complete. The Android file system could be mounted in OS of the cloud server and it could be checked for viruses. This has the advantage that the native instruction set is used and there is no additional overhead in emulating the smartphone's CPU. Therefore, the anti-virus security function has been placed in the native cloud OS for the proposed architecture, see section 5.1

6.4 CPU and battery consumption measurements in the Smartphone

The battery and CPU consumption in the smartphone for various security functions is one of the key factors that motivate the idea of offloading computation to the resource rich cloud environment. As a result of this, experiments were performed to understand the battery and CPU consumption of smartphones when performing virus scanning in various usage scenarios.

Battery and CPU monitoring widgets available in the Android market were used for monitoring. *Battery monitor widget* and *System panel task manager* are the applications that were used for battery and CPU measurements respectively. These widgets were chosen from a large variety of available widgets after careful examination for accuracy of the application by testing them in scenarios involving different CPU and battery consumption. Care was taken to ensure that the widgets themselves are not battery and CPU resource intensive. Detailed monitoring of the CPU usage was enabled to understand the utilisation of the CPU for different time periods. Kaspersky mobile anti-virus was chosen as it is one of the most well known and powerful anti-virus programs available for mobile phones. A HTC wildfire smartphone with Android version 2.1 was used for the following scenarios. The smartphone was configured with 1028 MB of data when the experiments were performed.

6.4.1 Virus scanning and 2 hour monitoring

Kaspersky anti-virus was started and a full scan of the smartphone file system was done and the CPU usage was monitored for a time period of 2 hours. The battery consumption was also monitored. Figure 18 shows that the CPU activity reached more than 80% during the scanning period. Figure 19 shows that the Kaspersky anti-virus process was in the top of the list of applications with respect to CPU usage. It is evident from the figure that this process has consumed upto 56.6% of CPU amounting to 1h 7m 57s of CPU usage during 2 hours, out of which the active scan time was 1h 16m 37s (or 63.8% of the 2 hours).

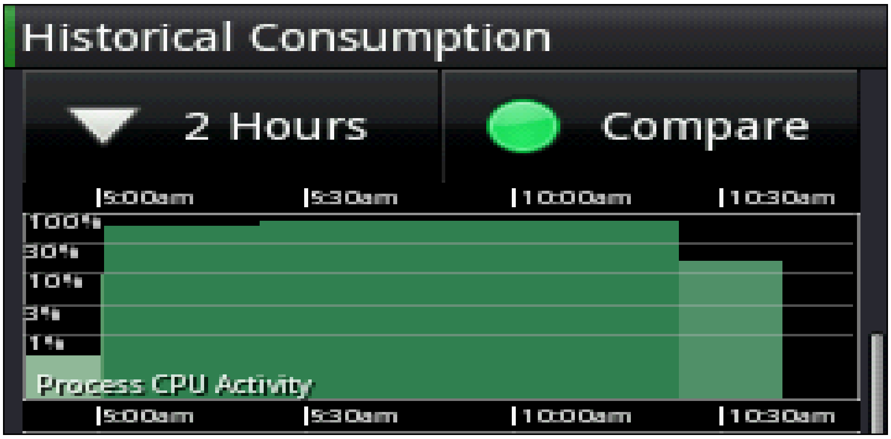
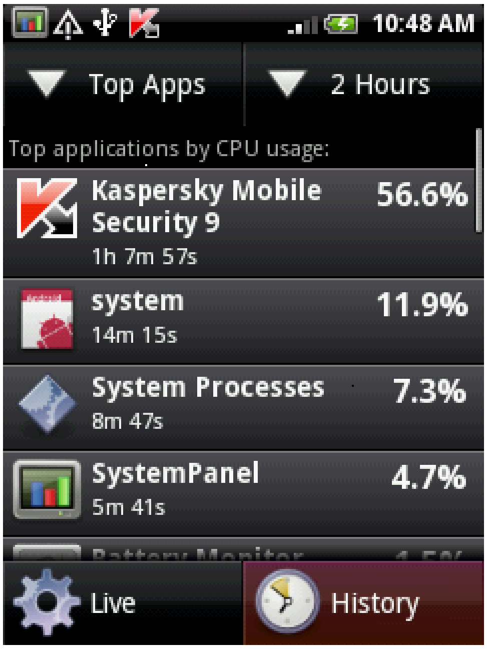


Figure 18: Kaspersky CPU consumption (2 hour monitoring)



Top applications by CPU usage:

Kaspersky Mobile Security 9 56.6%
1h 7m 57s

system 11.9%
14m 15s

System Processes 7.3%
8m 47s

SystemPanel 4.7%
5m 41s

Battery Monitor 4.5%

Figure 19: Top application classification by CPU usage for 2-hour time period

Figure 20 depicts the battery usage of the smartphone during the anti-virus scan versus time. The yellow mark in the graph represents the start of the anti-virus scan. When the anti-virus scan was started, the smartphone was fully charged and was in a stable state meaning that there were no fluctuations in the usage and a 100% charge was shown at the beginning of the battery monitor graph. Once the Kaspersky virus scan was started, the battery began to discharge at a median current of about 66 mA, but with wide fluctuations of upto 264mA drain. After completion of the scan the battery's capacity was reduced by about 8% leaving 92% of its fully charged capacity.

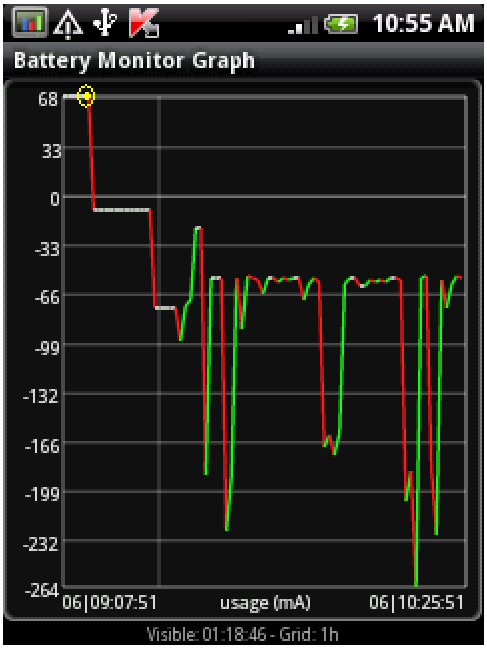


Figure 20: Battery current in mA vs. time for Kaspersky anti-virus

6.4.2 Virus scanning with one day monitoring

The Kaspersky ant-virus scan was performed once in 24 hours and one day's CPU usage of the smartphone was monitored while under normal usage to identify which applications used by a normal smartphone user are compute intensive. From the Figure 21, it can be seen that that very few applications that consumed more than 3% of the CPU; whereas virus scanning (whose duration is marked between the arrows) had a peak of almost 80% CPU consumption proving that it is extremely compute intensive. CPU usage is maximum for the Kaspersky mobile security application during a 24 hr time period when the smartphone was subjected to normal usage (playing music, connecting to the internet, watching online videos, etc.). This result is expected as the architecture of the smartphone is typically tuned to enable these normal activities to take place within the device's power budget.

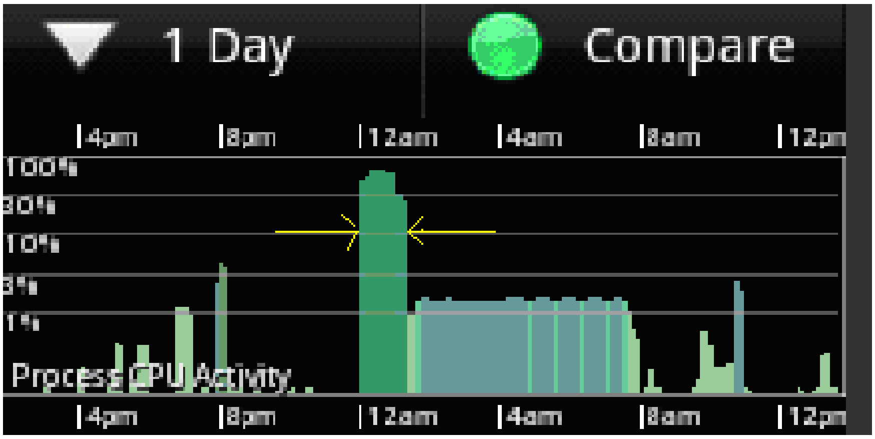


Figure 21: CPU activity in the smartphone over a time period of 24 hrs

6.4.3 Battery consumption when smartphone is on low battery

One of the results of the work presented in [35] is that the drain on the battery increases the energy expenses of the device for performing cryptographic operations. To understand if this fact holds true for virus scanning, the scan was started when the battery was low i.e around 13 % of it's fully charged state. The scan was performed over the same file storage as the previous scenarios. It was found that the battery power was reduced considerably by upto 11% of the available 13 % showing that performing the virus scan in a low battery state consumes more battery resources as previously illustrated in [35] for cryptographic operations.

7 Analysis of the proposed architecture

The architecture that has been proposed in chapter 5 is analyzed in this chapter in terms of its security aspects. Section 7.1 describes the security objectives taken into consideration in section 3.1 with respect to this architecture. Section 7.2 discusses the infection channel handled by this architecture. Section 7.3 describes the various security functions that can be provided as a service to the smartphones. Section 7.4 and 7.5 analyze the scalability and flexibility of the proposed architecture.

7.1 Security objectives

In this section we examine if the proposed solution meets the security objects as described earlier.

7.1.1 Confidentiality and Authenticity

In a smartphone usage scenario, confidentiality refers to maintaining the privacy of user's data in the smartphone. This is possible by requiring strong authentication for accessing the smartphone , for example requiring a strong password. This could be enforced by policy control provided as a cloud service for corporate smartphone users. The smartphone user also has the possibility to keep his/her file system encrypted for increased privacy. Further any personal information that is sent to the cloud server for utilizing the security services must be agreed upon by the user. The server also needs to authenticate itself to the user to enable a two-way authentication for establishing trust between the entities.

If the user would like to use the security functions as a one time service, then the user could be provided an option to not store their data in the cloud once the security function has completed execution.

7.1.2 Integrity

To protect the integrity of information, all connections used must be secure in nature in order to prevent any modification to the data in transit. For example, Man in the middle attack is possible, when the connection is not protected end to end. Suitable hashing techniques can be used in the cloud server to protect the integrity of the user’s data stored in the cloud.

7.1.3 Availability

Availability is highly dependent on the presence of an Internet connection and the battery capacity of the smartphone in order to support the smartphone's connection to the cloud server, as this is required to synchronize the user data. Furthermore, provision of cloud services to the smartphone user can be ensured by the use of back up servers in case of failure of the primary server.

7.1.4 Accountability

Monitoring and logging services set up in the cloud environment can be used to log the necessary data to keep track of activities in the cloud server. These logs can help determine the responsible entity in case of any security breach.

7.2 Infection Channel Considered

The targeted infection channel is the Internet connection. Some of the most common attacks against a smartphone occur through an active Internet connection. Since the services can be provided to the smartphone only with an active Internet connection, it is essential that the system handle attacks targeted against this infection channel. However, attacks via the Bluetooth link cannot be detected using the currently proposed architecture.

7.3 Security Functions

The security functions that have been listed below are some of the most relevant and obvious functions which can be offered to the smartphones. However, these do not represent an exhaustive list of functions.

7.3.1 Anti-virus

Virus scanning is considered one of the most compute intensive operations as it consumes a large amount of CPU and battery resources of the smartphone. This has also been proved by the measurements that have been carried out - as described in chapter 6. Cloud resources can be leveraged to perform the scanning and the results can be merged with the smartphone. Oberhiede et al.[8] present the idea of providing a virus scanning service in the cloud for smartphones and it has also been shown that detection coverage can be significantly improved by using multiple anti-virus programs in the virtual environment. In spite of these improvements, there are bandwidth constraints which need to be considered to decide if offloading actually can be beneficial. As in this architecture, the phone and the replica are kept synchronized, there is no need for transferring data *specifically* for virus scanning, thereby eliminating the bandwidth related problems. Virus scanning can be deployed in the native OS of the cloud server common to all replicas for improved performance and more effective utilization of resources.

7.3.2 Safe Browsing

The Internet is one of the prominent infection channels for malware. Smartphones are capable of keeping the users connected to the Internet most of the time through Wi-Fi or 3G networks. Safe browsing is one of the security functions that could be offered in the cloud. If the website accessed by the user is unsafe the system can notify the user. This is inspired from the F Secure Browsing protection for mobile devices which has been offered successfully as a cloud service [58]. Due to the availability of scalable storage resources in the cloud, this feature is easy to implement in the cloud server, common to all the smartphone replicas.

7.3.3 OS Integrity Checks

Integrity checks for the smartphone OS can be performed every time the smartphone is connected to the cloud server. The operating system files are synchronized to the cloud server thereby enabling the verification of the integrity of the smartphone's OS.

7.3.4 Remote Wiping and Versioning

A versioning system can be set up in the cloud server which can create snapshots of the smartphone's state and data at regular time intervals as specified by the user or the organization to which the user belongs. This can be significant by helping to safely restore the smartphone to a known state. It can also be used to recover a lost, corrupted, or infected smartphone to a clean state.

If the phone is misplaced, the cloud server can be used to remotely wipe the personal and corporate data present from the phone. Following this, suitable anti-theft or tracking software that is installed in the phone can be used to identify the phone's geographic location. Then, as described above, the versioning function can be used for recovery of the state of the phone, once it has physically been recovered.

7.3.5 Policy Control

In a corporate network, a policy control service can be utilized in a very efficient way to limit the actions of the smartphone users in the network. Various corporate policies could be specified which its users are expected to follow. Any violation of the policy could be effectively monitored by logging services. Examples of policies could include upload and download limits for a month, restrictions on the set of applications that can be installed on the smartphone, etc.

7.3.6 Secure Storage

Secure storage can be effectively provided as a cloud service to smartphones. In a corporate organization, confidential information and documents, if provided in a secure storage cloud, can be accessed by the employees when needed from their smartphones, instead of having them stored locally in their smartphones – potentially reducing corporate risk.

7.4 Scalability

Scalability of this architecture is highly dependent on the scalability of the cloud in the organization. Cloud computing in itself is known for its dynamically scalable resources available at a low cost that depends on the usage of the resources. For the initial performance tests discussed in chapter 6, the Amazon web services were used. An Amazon Virtual Private Cloud (VPC) could be exploited to establish a virtual private network taking complete control of the environment, thereby launching the Amazon resources inside this virtual network. Amazon VPC offers a wide range of functionalities and costs as little as $0.05 per connection hour in addition to the Amazon Elastic Compute Cloud (EC2) usage charges [56]. Table 7 shows the specification of a high CPU extra-large instance. A reserved Linux instance of this kind costs about $0.32 per hour considering the fact that there is a one-time payment for these kinds of instances, whereas an on-demand instance costs $0.76 per hour [57].

Consider the pricing for specific cloud services that can be offered to smartphones, for example anti-virus service. According to the measurements shown in section 6.4.1, the scan time for one full scan of the file system of the Android phone was 1h 16m 37s. So the total price to avail this service can be calculated by estimating the cost per hour for utilizing the cloud service. For example, using a High-CPU extra-large reserved Linux instance for the scanning would cost $0.64 ($0.32*2, since each partial instance-hour consumed will be billed as a full hour). But later on, for further scans, it would cost $0.32 as a full scan of the file system is not run each time.

Table 7: High-CPU Extra-large Amazon instance

7 GB of memory
20 EC2 Compute Units (8 virtual cores with 2.5 EC2 Compute Units each)
1690 GB of instance storage
64-bit platform
I/O Performance: High
API name: c1.xlarge

Instances are used to house the proxy, security functions and the smartphone replicas in the architecture. Smartphone OSs are much lighter when compared to the OSs of the PCs making virtualization comparatively easy. Furthermore, the display and transmission costs which are applicable for a mobile phone can be minimized when run in an emulated environment.

Additionally there could be some drawbacks when smartphone emulators are used as replicas, as the emulators are actually designed to function in a testing and debugging environment for the smartphones. Therefore, one should look at ways to optimize the smartphone emulators to work flawlessly in a replication scenario.

7.5 Flexibility

The security functions are additive in nature which implies that more functions can be added or existing functions that are not suitable could be removed. The security functions specified are only a few of the many that could be applied to this architecture.

8 Conclusions and Future Work

8.1 Conclusions

The aim of this thesis was to explore and identify the security functions that can be offered as a cloud service to smartphones. The motivation behind this research being the fact that the smartphones are not powerful enough in terms of battery and CPU power to support the various security functions that are needed for today's smartphone usage.

This thesis project examined the security requirements, current threats, infection channels, and limitations of smartphones in order to facilitate the identification of the security functions. The security requirements in terms of the security objectives such as confidentiality, integrity, availability, accountability were analysed. Recent threats to smartphones and the attack vectors for the same were identified. Infections through the Internet are more predominant when compared to the other attack channels. Furthermore the limitations of smartphones to support some of the compute intensive operations were studied. A deep analysis of architectures which integrate the mobile platform and the cloud computing paradigm was performed to understand how they can work together.

The results were used to propose a generic architecture "Security as a Service in Cloud for Smartphones". Since smartphone usage for official purposes and in corporate network has attracted more attention by posing significant risk, the architecture was tailored to suit its deployment in an organisation. The architecture used the idea of hosting replicas for smartphones in the cloud and executing the security functions on (or by) them. The various components and their functionalities needed for successful deployment of this architecture were detailed. Some of the security functions that can be offered in cloud were identified and included in the architecture.

The following are the key aspects of the proposed SeaaS architecture :

- Generic for all smartphone platforms
- Capable of providing various security services as a cloud offering as required
- Tailored to the needs of a corporate organisation, might be modified to be suitable for home users
- Handles threats that attack smartphones through the Internet
- Provides protection to smartphones only when they are connected to the Internet
- Offers flexibility in terms of adding and removing security functions

- The system operates transparent to the user to avoid any privacy breach, by keeping him/her informed of what data is being kept in the cloud.
- Energy consumption at the smartphone occurs only for the synchronisation process and it is common for all the security services

Since the motivation behind this idea was the resource constrained nature of smartphones, experiments with anti-virus scanning to compare the performance of smartphone and emulated smartphone were carried out. Battery power consumption for different use cases was also evaluated. The results obtained from the measurements showed that the emulated smartphone takes much less time than the actual smartphone to perform the virus scanning (as expected by using the abundant CPU resources in the cloud). Battery resources in each of the smartphone would also be used for anti-virus scanning in the smartphone, whereas scanning in the cloud uses the battery resources of smartphone only for synchronisation purposes.

8.2 Future Work

As a next stage to the proposal for the architecture, the most suitable way of achieving replication needs to be identified. Following that, a prototype will be implemented at Fraunhofer SIT. Each of the smartphone and cloud components needs to be designed and implemented to perform their stated functions. More security functions can be identified to be added to the list of cloud services for the smartphones. This requires more research as all security functions that can be provided to the smartphones cannot be provided as cloud service. This could not be carried out as part of the thesis project due to the limited time span.

The prototype implementation will be specific to a smartphone platform, so there is scope for research to understand the behaviour of this architecture for different smartphone operating systems and use cases. The security aspects should also be taken into account for the deployment of the prototype.

References

[1] Gartner Research, Accessed 3rd March 2011, http://www.gartner.com/technology/home.jsp

[2] Gartner Survey Shows U.S. Consumers More Likely to Purchase a Smartphone Than Other Consumer Devices in 2011, Gartner Press Release, February 17,2011, Accessed 27th June 2011. http://www.gartner.com/it/page.jsp?id=1550814

[3] Cisco 2010 Annual Security Report, January 2011, Accessed 27th June 2011 http://www.cisco.com/en/US/prod/collateral/vpndevc/security_annual_report_2010.pdf

[4] U.S. Federal IT Market Forecast 2011-2015, Market Research Media, September 2010 Update, Accessed 24th February 2011 http://www.marketresearchmedia.com/2009/05/23/us-federal-it-spending-forecast-2010-2015/

[5] Samsung Mobiles, Samsung Galaxy SII, Accessed 15th March 2011 http://galaxys2.samsungmobile.com/html/specification.html

[6] Security Guidance for Critical Areas of Focus in Cloud Computing V2.1, Cloud Security Alliance, December 2009, Accessed 13th February 2011 https://cloudsecurityalliance.org/csaguide.pdf

[7] B.-G.Chun, P.Maniatis, Augmented Smartphone Applications Through Clone Cloud Execution, In: Proceedings of the 12th conference on Hot topics in operating systems (HotOS 2009), May 18-20, 2009, Monte Verità, Switzerland

[8] J. Oberheide, K.Veeraraghavan, E.Cooke, J.Flinn, and F.Jahanian, Virtualized In-Cloud Security Services for Mobile Devices, In: Proceedings of the First Workshop on Virtualization in Mobile Computing (MobiVirt 2008), June 17-20, 2008, Breckenridge, Colorado, USA

[9] R. Chow, M. Jakobsson, and R. Masuoka, Authentication in the Clouds: A Framework and its Application to Mobile Users, In: Proceedings of the 2010 ACM workshop on Cloud computing security workshop (CCSW 2010), October 8, 2010, Chicago, Illinois, USA

[10] P. Stephanow and I. Tsvihun, Opportunities Of Security-as-a-Service On

Smart Phones, 25th Meeting of theWireless World Research Forum (WWRF 25), November 16-18, 2010, Newbury, UK

[11] G.Portokalidis, P.Homburg, K.Anagnostakis, and H.Bos, Paranoid Android: Versatile Protection For Smartphones, In: Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC 2010), December 6-10, 2010, Austin, Texas, US

[12] M. Memon, M. Hafner, and R. Breu, Security As A Service - A Reference Architecture for SOA Security. In: 7th International Workshop on Security in Information Systems (WOSIS 2009) at ICEIS 2009 Conference, May 6-7, 2009, Milan, Italy

[13] R.Kemp, N.O.Palmer, T. Kielmann, and H.E.Bal, Cuckoo: a Computation Offloading Framework for Smartphones, In: 2nd International Conference on Mobile Computing, Applications, and Services (MobiCASE 2010), October 25-28, 2010, Santa Clara, California, US

[14] E.Y. Chen, and M .Itoh, Virtual Smartphone over IP, In: Proceedings of the 2010 IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WOWMOM 2010), June 14-17, 2010, Montreal, Canada

[15] M .Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, The Case of VM-based Cloudlets in Mobile Computing, IEEE Pervasive Computing, 8, 14-23, October-December, 2009
doi: 10.1109/MPRV.2009.82

[16] G. Hogben and M. Dekker, Smartphones: Information security risks, opportunities and recommendation for users, December 2010, European Network and Information Security Agency (ENISA), Accessed 12th March 2011 http://www.enisa.europa.eu/act/it/oar/smartphones-information-security-risks-opportunities-and-recommendations-for-users

[17] F-Secure Labs, Worm: iPhoneOS/Ikee.B, Accessed 18th February 2011
http://www.f-secure.com/v-descs/worm_iphoneos_ikee_b.shtml

[18] F-Secure Labs, Trojan: SymbOS/ZeusMitmo.A, Accessed 1st March 2011
http://www.f-secure.com/v-descs/trojan_symbos_zeusmitmo_a.shtml

[19] Transaction Authentication Number, Wikipedia, Accessed 15th March 2011
http://en.wikipedia.org/wiki/Transaction_authentication_number

[20] Phising Attacks, Top ten Smartphone Risks, European Network and Information Security Agency (ENISA), Accessed 6th March 2011 http://www.enisa.europa.eu/act/application-security/smartphone-security-1/top-ten-risks/phishing-attacks

[21] D. Raywood, Google finds apparently fraudulent banking applications on its Android Marketplace, January 21, 2010, SC Magazine, UK, Accessed 1st March 2011

[22] US- Computer Emergency Readiness Team (CERT) Technical Information Paper – TIP-10-105-01, April 15, 2010, Accessed on 20th March 2011 http://www.us-cert.gov/reading_room/TIP10-105-01.pdf

[23] F-Secure Labs, Bluetooth-Worm: SymbOS/ Cabir, Accessed 26th February 2011 http://www.f-secure.com/v-descs/cabir.shtml

[24] F-Secure Labs, Worm: SymbOS/ Commwarrior, Accessed 26th February 2011 http://www.f-secure.com/v-descs/commwarrior.shtml

[25] N. Leavitt, Mobile Phones: The Next Frontier for Hackers?, Computer, 2005, 38, 20-23

[26] Samsung Mobiles, Samsung Wave II S8530, Accessed 20th March 2011 http://www.samsung.com/in/consumer/mobile-phone/mobile-phone/touch-phone/GT-S8530HKATHR/index.idx?pagetype=prd_detail&tab=specification

[27] F-Secure Labs, Trojan: SymbOS/Cardtrap.M, Accessed 26th February 2011 http://www.f-secure.com/v-descs/trojan_symbos_cardtrap_m.shtml

[28] F-Secure Virus Information Pages: Cxover.A, Accessed 10th March 2011 http://www.f-secure.com/v-descs/cxover_a.shtml

[29] F-Secure Labs, Mobile Security, Accessed 26th February 2011 http://www.f-secure.com/en/web/home_global/protection/mobile-security/overview

[30] Kaspersky Lab, Accessed 2nd March 2011 http://www.kaspersky.com/kaspersky_mobile_security

[31] BlackBerry Enterprise Solution: Security Technical Overview, Version 5.

Accessed 27th February 2011 http://docs.BlackBerry.com/en/admin/deliverables/7127/BB_Ent_Soln_Security_5_0_0_STO.pdf

[32] M. Jakobsson, E. Shi, P. Golle, and R. Chow, Implicit Authentication for Mobile Devices, In: 4th USENIX Workshop on Hot Topics in Security (HotSec '09), August 11, 2009, Montreal, Canada

[33] M .Nauman and T. Ali, TOKEN: Trustable Keystroke-Based Authentication for Web-Based Applications on Smartphones, Information Security and Assurance: Communications in Computer and Information Science, 2010,76, 286-297, Springer, Berlin/Heidelberg, ISBN: 978-3-642-13365-7 http://dx.doi.org/10.007/978-3-642-13365-7_28

[34] Y. Shin, M. Gupta, and S. Myers, A Study of the Performance of SSL on PDAs, In:12th IEEE Global Internet Symposium (GI 2009), April 24, 2009, Rio de Janerio, Brazil

[35] H.Rifà-Pous and J.Herrera-Joancomarti, Computational and Energy Costs of Cryptographic Algorithms on Handheld Devices, Future Internet, ISSN 1999-5903, 3(1): 31-48, 2011, doi:10.3390/fi3010031, http://www.mdpi.com/1999-5903/3/1/31/

[36] A.D. Zayas and P.M. Gomez, A Testbed for Energy Profile Characterization of IP Services in Smartphones over Live networks, Mobile Netw. Appl., 2010, **15**, 330-343

[37] A. Carroll and G. Heiser, An Analysis of Power Consumption in a Smartphone, In: Proceedings of the 2010 USENIX Annual Technical Conference (USENIXATC'10), June 23-25, 2010, Boston, Massachusetts, USA

[38] Opera Mini, Wikipedia, Accessed 13th March 2011 http://en.wikipedia.org/wiki/Opera_Mini

[39] Opera Software, Accessed 7th March 2011 http://www.opera.com/mobile/help/faq/#security

[40] M. Memon, M. Hafner, and R. Breu, SECTISSIMO - A Platform-independent Framework for Security Services, In: Modeling Security Workshop in Association with MODELS '08, 28th September, 2008, Toulouse, France

[41] N. Dragoni and F. Massacci, Security-By-Contract (SxC) for Mobile

Systems – or how to download software on your mobile without regretting it, Telektronikk Journal, **1**, 107-116, 2009, Telenor

[42] M.D. Lopez, Research Report: Successful Mobile Deployments Require Robust Security, May 25, 2009, Accessed on 20th February 2011 http://us.BlackBerry.com/ataglance/get_the_facts/Successful_Mobile_Deployments.pdf

[43] Feature and Technical Overview: BlackBerry Enterprise Server for Microsoft Exchange, Version 4.1, Service Pack 5, Accessed on 2nd March 2011 http://docs.BlackBerry.com/en/admin/deliverables/1397/Feature_and_Technical_Overview.pdf

[44] BlackBerry Enterprise Solution: Security Technical Overview, Version 5. Accessed 27th February 2011 http://docs.BlackBerry.com/en/admin/deliverables/7127/BB_Ent_Soln_Security_5_0_0_STO.pdf

[45] BlackBerry Wireless data Security Features, Accessed 24th February 2011 http://uk.BlackBerry.com/ataglance/security/features.jsp

[46] A.Azfar, Multiple Escrow Agents in VoIP, Masters Thesis, School of Information and Communicatin Technology, Royal Institute of Technology (KTH), Stockholm, Sweden, TRITA-ICT-EX-2010:109, June 2010 http://web.it.kth.se/~maguire/DEGREE-PROJECT-REPORTS/100607-Abdullah_Azfar-with-cover.pdf

[47] J.Sundgot, Smartphone virus gets clever with social engineering, infoSync World, 21 October 2005, http://www.infosyncworld.com/news/n/6252.html

[48] E.Karamanos, Investigation of home router security, Masters Thesis, School of Information and Communicatin Technology, Royal Institute of Technology (KTH), Stockholm, Sweden, TRITA-ICT-EX-2010:38, April 2010 http://web.it.kth.se/~maguire/DEGREE-PROJECT-REPORTS/100411-Emmanouil-Karamanos-with-cover.pdf

[49] J.-E.Ekberg, and Markku Kylänpää, Mobile Trusted Module (MTM) - an introduction, Nokia Research Center, Helsinki, Finland, NRC-TR-2007-015, November 14, 2007, http://research.nokia.com/files/tr/NRC-TR-2007-015.pdf

[50] T.K.Buennemeyer, M.Gora, R.C.Marchany, and J.G.Tront, Battery

Exhaustion Attack Detection with Small Handheld Mobile Computers, IEEE International Conference on Portable Information Devices, March 2007, ISBN: 1-4244-1039-8, doi: 10.1109/PORTABLE.2007.35, http://filebox.vt.edu/users/gora/public/files/Buennemeyer_Gora_Portable07.pdf

[51] One hundred million, Opera Press Release, Accessed 1st April 2011 http://www.opera.com/press/releases/2011/02/10/

[52] Opera reaches (another) 100 million users, Opera Press Release, Accessed 12th April 2011 http://www.opera.com/press/releases/2011/04/07/

[53] B.-G.Chun, S.Ihm, P.Maniatis, M.Naik, and A.Patti, Elastic Execution between Mobile Device and Cloud, In: Proceedings of the 6th Conference on Computer Systems (EuroSys'11), April 10-13, 2011, Salzburg, Austria

[54] J.Twentyman, Lost smartphone pose significant corporate risk, Secure Computing Magazine for IT Security Professionals, 12th Feb 2009, Accessed 7th April 2011 http://www.securecomputing.net.au/Feature/137038,lost-smartphones-pose-significant-corporate-risk.aspx

[55] B.Zhao, Z.Xu, C.Chi, S.Zhu, and G.Cao, Mirroring Smartphones for Good: A Feasibility Study, ZTE Communications, 18th March 2011, Accessed 3rd May 2011 http://wwwen.zte.com.cn/endata/magazine/ztecommunications/2011Year/no1/articles/201103/t20110318_224543.html

[56] Amazon Virtual Private Cloud (Amazon VPC), Accessed 5th May 2011 http://aws.amazon.com/vpc/

[57] Amazon Elastic Compute Cloud (Amazon EC2), Accessed 5th May 2011 http://aws.amazon.com/ec2/#pricing

[58] F-Secure Labs, Pressroom: F-Secure boosts smartphone security with Browsing Protection and Locator-feature, Accessed April 25th 2011 http://www.f-secure.com/en_IN/about-us/pressroom/news/2010/fs_news_20101502_02_eng.html

[59] AVG Mobilation for Android Smartphones & Tablets, AVG Home Security, Accessed 6th May 2011, http://www.avg.com/de-en/antivirus-for-android

[60] NetQIn Mobile Anti-virus for Android, Accessed May 2nd 2011 http://www.netqin.com/en/antivirus/

[61] C.D.Sanchez, Speaker Recognition in a handheld Computer, Masters Thesis, School of Information and Communication Technology, Royal Institute of Technology (KTH), Stockholm, Sweden, TRITA-ICT-EX-2010:285, November 2010, http://web.it.kth.se/~maguire/DEGREE-PROJECT-REPORTS/101117-Carlos_Dominguez-with-cover.pdf

[62] Clam AV, Accessed June 28th 2011 http://www.clamav.net/lang/en/