Spring 4-2015

REMOTE MOBILE SCREEN (RMS): AN APPROACH FOR SECURE BYOD ENVIRONMENTS

Santiago Manuel Gimenez Ocano
University of Nebraska-Lincoln, sxntixgo@gmail.com

REMOTE MOBILE SCREEN (RMS): AN APPROACH FOR SECURE BYOD ENVIRONMENTS

by

Santiago Gimenez Ocano

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Byrav Ramamurthy

Lincoln, Nebraska

April, 2015

REMOTE MOBILE SCREEN (RMS): AN APPROACH FOR SECURE BYOD ENVIRONMENTS

Santiago Gimenez Ocano, M.S.

University of Nebraska, 2015

Adviser: Byrav Ramamurthy

Bring Your Own Device (BYOD) is a policy where employees use their own personal mobile devices to perform work-related tasks. Enterprises reduce their costs since they do not have to purchase and provide support for the mobile devices. BYOD increases job satisfaction and productivity in the employees, as they can choose which device to use and do not need to carry two or more devices.

However, BYOD policies create an insecure environment, as the corporate network is extended and it becomes harder to protect it from attacks. In this scenario, the corporate information can be leaked, personal and corporate spaces are not separated, it becomes difficult to enforce security policies on the devices, and employees are worried about their privacy. Consequently, a secure BYOD environment must achieve the following goals: space isolation, corporate data protection, security policy enforcement, true space isolation, non-intrusiveness, and low resource consumption. We found that none of the currently available solutions achieve all of these goals.

We developed Remote Mobile Screen (RMS), a framework that meets all the goals for a secure BYOD environment. To achieve this, the enterprise provides the employee with a Virtual Machine (VM) running a mobile operating system, which is located in the enterprise network and to which the employee connects using the mobile device. We provide an implementation of RMS using commonly available software for an x86 architecture.

We address RMS challenges related to compatibility, scalability and latency. For the first challenge, we show that at least 90.2% of the productivity applications from Google Play can be installed on an x86 architecture, while at least 80.4% run normally. For the second challenge, we deployed our implementation on a high-performance server and run up to 596 VMs using 256 GB of RAM. Further, we show that the number of VMs is proportional to the available RAM. For the third challenge, we used our implementation on GENI and conclude that an application latency of 150 milliseconds can be achieved.

ACKNOWLEDGMENTS

To my advisor, Dr. Byrav Ramamurthy, for his continued support, encouragement and guidance. Being his student and advisee has been inspiring and a privilege from the very first day. My research topic was suggested by him while I was a student in his Data and Network Security class, which was the most influential class I have taken.

I would also like to thank Dr. Yong Wang, from Dakota State University, for his continuous help and advice on my research. His suggestions and comments helped my research greatly.

I am also thankful to the members of my MS Thesis committee, Dr. Witawas Srisa-an and Dr. David Swanson for their service and reviewing my thesis. Their feedback and suggestions helped me improve this work.

A note of thanks to the members of the Networking Lab. I would specially thank Adrian Lara, Sara El Alaoui and Pan Yi. I would also like express my gratitude to Rigoberto Wong for all his help.

Part of this work was performed on computers from the Holland Computing Center. I am thankful to its director, Dr. David Swanson, and Garhan Attebury for allowing me the use of these assets as well as for helping me.

Finally, I would like to thank my love, Nicole, my parents and my sister for their love, support and understanding.

This research was partially supported by a National Science Foundation (NSF) Grant No. CNS-1345277.

My education at UNL was supported by the Argentinian Presidential Fellowship coordinated by Fulbright Commission in Argentina and Jefatura the Gabinete de Ministros de la Republica Argentina.

Contents

List of Figures

List of Tables

List of Acronyms

2TAC 2-Tier Access Control.

BSF BYOD Security Framework.

BYOD Bring Your Own Device.

EEM Enterprise Mobility Management.

MAM Mobile Application Management.

MDM Mobile Device Management.

MIM Mobile Information Management.

MVM Mobile Virtual Machine.

MVP Mobile Virtualization Platform.

NAC Network Access Control.

NAT Network Address Translation.

OS Operating System.

PUA Potential Unwanted Applications.

RBAC Role-Based Access Control.

RDP Remote Desktop Protocol.

REE Rich Execution Environment.

RFB Remote Framebuffer.

RMS Remote Mobile Screen.

SSA Security Service Architecture.

TEE Trusted Execution Environment.

TPM Trusted Platform Module.

VM Virtual Machine.

VMM Virtual Machine Monitor.

VNC Virtual Network Computing.

VPN Virtual Private Network.

Chapter 1

Introduction

Mobile devices have become integral elements in our daily life, and they have become ubiquitous. For example, in 2013, the adoption of mobile devices and connections grew to 7 billion units, according to a report from Cisco [1]. To put this figure into perspective, according to the United Nations there are 7.2 billion inhabitants in the world [2].

Mobile devices have had a tremendous impact on companies, since they increase the productivity of the employees, as well as provide flexibility in terms of time and space. Consequently, companies have been providing their employees with mobile devices to enable them to perform their job-related tasks. However, the extensive use of these devices has created inconveniences for enterprises since they must handle the costs associated with obtaining and maintaining such devices. Additionally, the incredible speed in which new technologies are introduced make the current models of these devices less appealing to the employees after a short period of time. This results in situations where employees want to change their devices faster than the enterprises can provide them with new ones.

As a result of this choice and customization in mobile devices, employees often

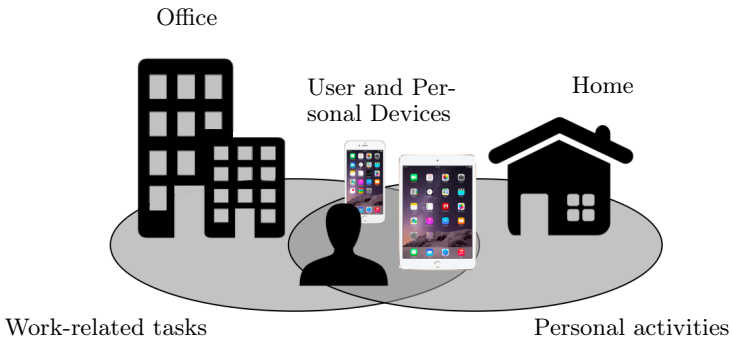

Figure 1.1: Representation of a BYOD environment.

request their companies to allow them to use their personal mobile devices for work-related tasks while also retaining them for personal use [3]. Because of this merging of usage, these devices are known as dual-use devices [4].

1.1 Description of BYOD

In these environments, companies have adopted solutions in the form of new policies. This set of policies is known as Bring Your Own Device (BYOD), which allows an employee to use the mobile devices they prefer to perform work-related tasks. In a recent study carried out by Cisco, it was found that 89% of IT departments enable BYOD in some form [5]. A typical BYOD environment is depicted in Figure 1.1, where an employee uses a personal smartphone and a personal tablet not only for personal activities but also for work-related tasks.

BYOD provides a series of advantages for both employees and the enterprise, which are described below:

1.1.1 Job Satisfaction

The use of BYOD policies produces an increase in job satisfaction in the employees. As mentioned before, employees can select the device they feel comfortable with and replace it at the time of their choosing. They also avoid carrying additional devices by using a single device for both personal and work uses. A Cisco report [5] mentions that the main reasons for employees to use personal devices in a BYOD scenario are the "any device work style", the possibility of combining work and personal activities, the avoidance of usage restrictions, the "consumer experience" at work, and the access to personal mobile applications.

1.1.2 Productivity

After applying BYOD policies, companies have seen an increase in productivity in terms of output as well as an increment in collaboration among employees. According to Cisco [5], this is the most important finding in their research, since it shows that the use of personal devices in the BYOD environment does not result in distractions, but it has the opposite effect. Further, Assing et al. [6] mentioned that staff productivity is increased by the fact that mobility allows the employees to be productive from anywhere.

1.1.3 Recruitment

Loose et al. [7] showed that there is a highly significant correlation between how attractive an enterprise is to a future employee and the adoption of BYOD by that enterprise. Hayes [8] supports this concept by showing that a company that applies BYOD policies is more appealing to potential employees, as the enterprise is seen as a flexible work environment.

1.1.4 Cost Saving

Assing et al. [6] state that BYOD has its origins in the fact that companies wanted to reduce costs of equipment available to employees by letting them purchase the mobile devices they desire. Additionally, enterprises do not have to incur in expenses related to providing technical support for such devices. Moreover, technical support resources can be either focused on a better service, allocated to other area of the enterprise, or simply reduced.

1.2 Contribution

This thesis addresses the following question:

Can a BYOD solution be implemented in such a way that it provides security to the corporate data as well as privacy to the employee?

In order to answer this question, this work provides the following contributions:

- In order to achieve a secure BYOD environment, we extend the goals previously found in literature by including true space isolation, non-intrusiveness and low resource consumption.
- We provide a comprehensive survey of all the solutions for BYOD environments and we describe why they do not achieve all the goals needed for a secure BYOD environment.
- We provide a framework that meets all the required goals for a secure BYOD environment. We provide a proof-of-concept and evaluate our solution.

These contributions are part of the paper "Remote Mobile Screen (RMS): an approach for secure BYOD environments" [9] presented at the International Conference on Computing, Networking and Communications (ICNC) 2015, technically sponsored by IEEE, and of a survey paper submitted to a journal [10]. Finally, a paper related to the implementation challenges of RMS is under review [11].

1.3 Outline of Thesis

The rest of this thesis is structured as follows. Chapter 2 provides a background on security related to mobile devices, the challenges to provide a secure BYOD environment and the goals that must be met in order to overcome these challenges. In Chapter 3 we provide a comprehensive classification and description of the current solutions for secure BYOD environments, either commercial solutions or from the research community. We analyze these solutions and show that none of them achieve all the necessary goals. Remote Mobile Screen (RMS) and its architecture is presented in Chapter 4. In this chapter, we analyze the features offered by RMS, we show an implementation of this framework, we discuss the challenges presented by this solution, and we show the results of evaluations related to scalability and latency. In Chapter 5 we provide the conclusion of this thesis, as well as future research directions for this work. Appendix A presents the list and description of the mobile applications evaluated in our usability test. In Appendix B we provide the scripts used for our scalability evaluation. Finally, the scripts used for our latency evaluation are provided in Appendix C.

Chapter 2

Background

As described in Chapter 1, BYOD policies present great benefits for both the enterprise and the employee. However, the implementation of these policies creates threats related to the corporate information and the privacy of the employee. In this Chapter we present the threats on mobile devices, we show that these threats create challenges for a secure BYOD environment, and we describe a set of goals that must be met to overcome these challenges.

2.1 Threats on Mobile Devices

These types of threats focus on exploiting vulnerabilities found in the architecture of the mobile device. These attacks are generally in the form of malware or vulnerabilities of the communication capabilities. Consequently, we present the flow of an attack on mobile devices, a description and classification of malware, and the vulnerabilities found in the communication sub-layer.

2.1.1 Flow of Attacks on Mobile Devices

Wang et al. [12] describe how an attack is performed on mobile devices. It starts by the attacker exploiting a vulnerability through an attack vector, which generally targets the application layer, then the attack goes through the communication layer to reach the resource layer. Then, the flow of the attack goes all the way back through the layers to a premium account or malicious website. Finally, a loop is formed when the flow of the attack reaches back the attacker. Figure 2.1 shows the process involved in an attack on a mobile device.

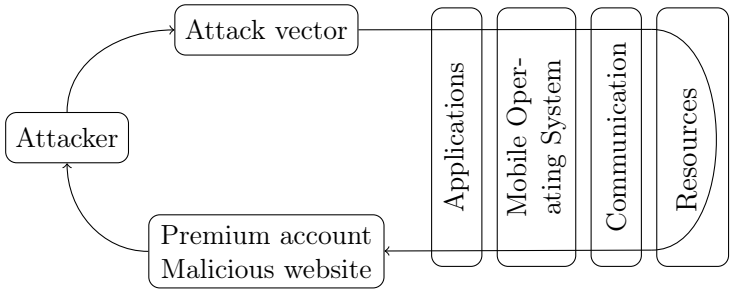

Figure 2.1: Flow of a malware attack on mobile devices (based on [12]).

2.1.2 Mobile Malware and Potential Unwanted Applications (PUA)

Mobile malware and PUA are the main concerns in mobile devices. According to a report from F-Secure Corporation [13], malware is software whose objective is to pose security risks that affect the user's information or system. Malware is classified into the following categories:

- Backdoor: it is a program that allows unauthorized access to information by avoiding authentication mechanisms; in the case of mobile devices, this unauthorized access is remote.

- Trojan Horse: it is an application that pretends to be benign but whose real objective is to affect negatively the user's system and data.
- Worm: it is software whose goal is to produce an exact replication of itself.

A PUA is a piece of software that can be considered unwanted or intrusive by the user, and that introduces privacy or security risks. If the user knows about such risks, he or she can accept them and decide to install the application. The different types of PUAs are listed below:

- Spyware: it is an application that obtains information about the user's system and behavior and then stores that information for future analysis by an attacker.
- Trackware: it is a program that collects data that can be used to identify and locate the user or the device by a third party.
- Adware: it is an application that shows advertisement information in an unsolicited way or that might expose the user to privacy or security risks.

The main problem with malware in BYOD environments is that mobile devices might spread out these malware over the corporate network, infecting other mobile devices and compromising the corporate resources. This can affect the productivity of the employees, but it can also lead to bigger security breaches, such as unauthorized access to confidential corporate information.

Following the high adoption of Android and iOS, malware developers have been targeting the vulnerabilities found in these mobile Operating System (OS)es. The number of malware has been increasing over the years, Android being the main target. According to statistics from F-Secure Corporation [13], in the first quarter of 2013 there were 166 new mobility threats, 153 of them targeting Android devices; by the

third quarter of the same year, there were a total of 259 new mobility threats, 252 being Android-specific. By the first quarter of 2014, there were 277 new mobile threats identified, 275 of these threats targeted the Android platform [14].

The open-source philosophy behind Android is the reason why this mobile OS has become the main target of malware. Android allows users to install applications from different marketplaces, which generally present more malware [15]. Additionally, the process used to publish applications in Google Play did not include any type of pre-verification in the publishing procedure. This way, it is the user who had to make sure that the application is free of any malware before installing it [16]. Recently, Google changed this policy and now the mobile applications are verified before becoming available in Google Play. [17].

2.1.3 Phishing

Phishing is a form of social engineering, which generally starts by e-mail, where the user is led to a website that pretends to be a trusted one. The objective of this attack is to have the user reveal personal information that can be used later. For example, one of the most common phishing attacks is where the user receives an e-mail from a bank. This e-mail contains a link to a fake e-banking website, where the user is asked to type his or her credentials.

Trend-Micro reports two main reasons why mobile devices are being target of phishing. First, the fact that mobile web browsers are not able to display enough information related to security elements. Second, the fact that all mobile device comes with a web browser that opens by default after a link has been clicked [18]. The same report mentions that 75% of the phishing attacks have the objective to gather information related to financial activities.

As an example of a phishing attack, an employee can be led to reveal his or her corporate credentials, which in turn can be used to access all the content to which the employee has been granted access. Based on the role of the employee, this could lead to a massive leakage of secret corporate information.

2.1.4 Theft, Loss and Sale

Given the fact that mobile devices are small and portable, they can be easily stolen or lost. A report from the Office of the New York State Attorney General [19] mentions that in 2013 nearly 3.1 million of mobile phones were stolen in the US. Stolen mobile devices must be a concern for any enterprise. If the employee stored sensitive corporate data on the mobile device, and if the device is not properly protected, that data can be accessed by an unauthorized party.

In The Symantec Smartphone Honey Stick Project [20], the security company intentionally lost 50 smartphones containing simulated corporate and personal data, along with monitory tools to analyze how the smartphones were used after they were found. The main goal behind this experiment was to inform the owners about what to expect once their smartphones were lost. Their two main findings are: (1) it is highly likely that a stranger will attempt to access personal and business information if the phone is unprotected, and (2) the owner of the phone should not expect to be contacted by the person who found the phone. Detailed statistics of their findings can be found in Table 2.1 for corporate data, and in Table 2.2 for personal data. We can see that in either corporate or personal data, the percentage of access is at least 40% for each of the categories, which indicates that an acceptable level of privacy or confidentiality cannot be expected once a mobile phone has been lost.

Another important situation that presents a threat for corporate information is

Type of data	Percentage
General	83
E-mail	45
HR-related files	40-53
Remote Admin app	49

Table 2.1: Percentage of corporate accessed data (based on information provided by [20].)

Type of data	Percentage
General	89
Photos	72
Password reset attempt	66
Social networking	60
E-mail	60
"Saved Password" file	57
Online banking app	43

Table 2.2: Percentage of personal accessed data (based on information provided by [20].)

when employees sell their mobile devices to a third party. In this case, if the information contained in the device has not been properly erased, corporate data can be leaked. According to a study from CPP in 2011 [21], 54% of the second hand smartphones sold contain extensive personal data which includes credit and debit card PIN numbers, bank account details, passwords, phone numbers, company information and log in details to social networking sites like Facebook and LinkedIn. In that study 81% of the previous owners claimed that the data was properly deleted, with 60% being confident that all the personal data was removed.

AVAST Software conducted an experiment that showed the risks related to selling a mobile device [22]. In this experiment, the researchers purchased from ebay 20 Android smartphones that supposedly had all their data erased. From that set of devices, the researchers were able to recover more than 40,000 photos, 750 e-mails and text messages, and 250 contacts. Additionally, the security firm was able to obtain the identity of the previous owner, and in one case one complete loan application was

retrieved. According to AVAST, 80,000 used smartphones are listed for sale online every day.

2.1.5 Extension of the Network

Wang et al. [23] describe the concept of BYOD devices and how they extend the corporate network. Once a mobile device connects to the corporate network from outside of the enterprise perimeter, for example using a Virtual Private Network (VPN), it can be said that the device extends the size of the network, as the latter geographically increases its range. This creates a threat since network administrators and security officers cannot efficiently control this increment in the size of the network, which in turn enables the possibility of attacks from unprotected or compromised devices. Further, since mobile devices can be used as Wi-Fi hotspots, they can provide outsiders access to the corporate network [24]. Consequently, in order to achieve an acceptable level of security in the corporate network, it is extremely important to have the BYOD devices to comply with all security policies and good practices set by the enterprise.

2.1.6 Insider Attack

Salem et al. [25] provide an excellent survey on insider attacks detection. This survey introduces the concept of this type of attack by giving an example of a compromised cell phone. The authors provide evidence that insider attacks have become a bigger threat than malware. Since the security countermeasures are focused on unauthorized and illegitimate access to the corporate network, threats that result from the misuse of devices inside the network represent the biggest source of most damaging malicious activities. According to this survey, there are two types of malicious insiders: traitors

and masqueraders. While the former is an employee that has authorization and can access the company assets, the latter is an employee that steals the identity (and the privileges) of a valid user in order to execute malicious activities.

In the context of BYOD environments, it is extremely important to secure the network against insider attacks, as the mobile devices are authorized to connect to the corporate network. If no protection against these types of attack is deployed, it becomes trivial for any malware to perform malicious activities despite the intentions of the employee.

2.1.7 Compromised Corporate Network Attacking Mobile Devices

So far, we have discussed the problems where the use of mobile devices can compromise the security of the corporate network. However, it is imperative to also consider the case where an already compromised network threats mobile devices, as in this case the devices are also used for personal activities. In other words, we need to consider the problem from the employee's perspective, since he or she could connect a personal device to a network that might contain malware, compromising the employee's personal data. It becomes interesting to explore the behavior of the enterprise and the position it adopts under this type of situations, as the enterprise must detect and eliminate any type of threat to the devices. Otherwise, employees would be discouraged from making use of the BYOD policies, which in turn would reduce their productivity and job satisfaction.

2.2 Security Challenges Related to BYOD Environment

Based on the threat model presented in the previous section, we can introduce a set of challenges that a secure BYOD environment present. While the first three challenges were identified by Wang et al. [23], the last one was discussed by Miller et al. [26]. We present these four challenges as follows.

2.2.1 Data leakage

We categorize the corporate data as public or confidential. In the first category, the information is freely distributed by the enterprise. However, for the second category, the enterprise spends resources to prevent data leakage.

In a BYOD environment, employees have access to the confidential information through their mobile devices. Considering all the threats that affect mobile devices, there are challenges related to how to secure the corporate information once it reached such devices.

2.2.2 Unauthorized sharing of spaces

In BYOD environments we can define two spaces: a personal space and a corporate space. On the one hand, the personal space includes all applications and documents owned by the employee such as family photos, personal contacts, or leisure applications like games. On the other hand, the corporate space includes all the applications and documents related to the enterprise, such as corporate emails and contact lists as well as productivity applications provided by the enterprise, like a spreadsheet editor.

The challenge in securing BYOD environments is how to keep these two spaces

isolated from each other. In other words, to provide mechanisms to prevent the access of personal data from enterprise-related tasks, and vice versa.

2.2.3 Lack of security compliance

In many BYOD scenarios the enterprise finds it difficult to enforce its security policies due to the fact that the employees are the owners of the mobile devices. For example, if the policies state that mobile devices must run antiviruses to prevent malware, the enterprise must check that all devices comply with this directive. Further, testing device-by-device is not an option since it is time consuming and does not scale well.

2.2.4 Employee privacy

There is concern related to employee's privacy, as companies might monitor the employee's personal activities as well as analyze his or her personal information. This is specially a concern when he or she is connected to the corporate network, as the later could potentially track all the data in the network. Consequently, the employee might not feel comfortable using his or her mobile device, which negatively affects productivity and job satisfaction, and defeats the purpose of BYOD policies.

2.3 Goals for a Secure BYOD Environment

Considering the challenges described in section 2.2, a set of goals can be defined for secure BYOD environments. Wang et al. [23] have described the first three goals in the following list, while the remaining three have been identified by Gimenez Ocano et al. [9]. The latter authors stated that the first set of goals are necessary but not sufficient to achieve the goal of a secure BYOD environment, since they do not consider the resource constraints of the mobile devices, the privacy invasion that a

employee might experience, and other situations that the first set of goals does not address.

2.3.1 Space Isolation

This goal addresses the challenge of unauthorized sharing of spaces. This is achieved by isolating the personal space from the corporate space in such a way that no data can be sent from one space to another.

However, the implementation of space isolation does not prevent situations where the enterprise performs a restoration to factory default configuration, with the purpose of deleting all confidential information located in the device. This is not desirable because, under these circumstances, the employee would lose all the data saved in the personal space.

2.3.2 Corporate Data Protection

The confidential information of the enterprise must remain secret even after a mobile phone is sold, lost or stolen. In this way, only authorized employees can access the information. Cryptographic algorithms can be used to cipher the corporate data such that only the employee that has the key can access such data.

2.3.3 Security Policy Enforcement

Since policy enforcement is hard to achieve for mobile devices, one of the goals is to make this enforcement automatic through the use of software. Moreover, because it results unfeasible to check each mobile device at the time, policy enforcement must be performed through automatic checkups based on software.

Additionally, a solution should include mechanisms to translate policies into software configuration that comply with them.

2.3.4 True Space Isolation

Even though space isolation is enough to avoid data sharing between the personal space and the corporate space, corporate data should not be saved in mobile devices. This goal addresses situations where the mobile device is sold, lost or stolen and, even though there is space isolation and data protection, the data could still be theoretically recovered by cracking the encryption algorithm.

With true space isolation, this issue can be addressed by storing the corporate data only in corporate resources, while preventing the personal data to be sent to the enterprise. In this way, even if the mobile device is sold, lost or stolen, the corporate data remains protected.

2.3.5 Non-intrusiveness

From the employee's perspective, a BYOD solution should not require any privileged permissions on mobile devices, as the employee can be suspicious about invasion of privacy. Agents or monitoring applications should also be avoided because of the same reasons. Even if the installed agent does not invade the personal space, the employee might be suspicious.

Additionally, solutions should not modify the operating system or use a custom ROM that disrupts mobile devices' original configuration. This type of modification can be against the terms of use, as it is shown in section 2(d) of the iOS License Agreement provided by Apple [27], and void the guaranty of the mobile device. Further, this type of solution requires manually saving the original configuration, and

then installing and uninstalling such custom mobile OS by the IT department, which does not scale well. Moreover, there might be situations where the employee leaves the enterprise and could not recover the original configuration.

2.3.6 Low Resource Consumption

Any solution should consider the scarcity of resources proper of mobile devices. This means that solutions that require the use of heavy processing or RAM memory can affect the usual behavior of the mobile device, which might lead to a decrease of adoption from the employees.

In this chapter we have provided a background related to the threats that mobile devices present, the security challenges related to BYOD environments, and the necessary goals for a secure BYOD environment. In the following chapter we will discuss the solutions available for BYOD scenarios. We will classify them according to their type and we will discuss the goals they achieve.

Chapter 3

Related Work

There are different solutions used in BYOD environments. These solutions provide different features and address some of the goals listed in Section 2.3. We can classify them based on their type in MVM, Agent-based, Cloud-based, VPN, Trusted Environment and Framework. In general, solutions that are in the same category achieve the same set of goals. However, solutions classified as framework might not meet the same goals. In Figure 3.1 we provide a detailed taxonomy of the solutions for BYOD environments that will be covered in the rest of the section.

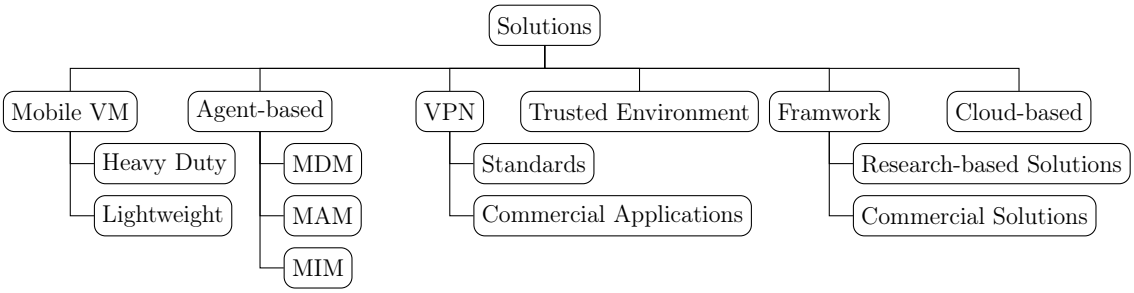

Figure 3.1: Taxonomy of BYOD solutions.

3.1 MVM

The use of VMs for separating the personal space from the corporate space is discussed by Wang et al. [23], with references to the work of Andrus et al. [28] and Barr et al. [29]. Mijat et al. [30] provide a good description of how virtualization works. In order to offer virtualization, a thin control software called Virtual Machine Monitor (VMM) is used between the OS and the hardware. Then, the VMM can run with privileged permissions and can host different OSes, which in turn run in a sandbox denoted VM. The VMM coordinates the access from the OSes to the hardware in a way that the OSes can use the hardware resources as if they had exclusive access to them. These OSes are called guest OSes.

There are two types of virtualization: full virtualization and paravirtualization. In the first case, the guest OS does not know that it is part of a virtualized environment and it does not require any type of addition or modification. Full virtualization provides the best solution if isolation is required, but it is at the expense of extra resource requirements and complexity in the VMM. Paravirtualization is achieved by modifying the guest OS in a way they have direct access to the VMM. This way, the complexity and requirements for implementing a VM environment can be decreased.

An example of full virtualization for mobile phones is INTEGRITY Multivisor [31], which has been implemented in mobile phone architectures by supporting ARM TrustZone [32]. However, most of solutions in for mobile devices are based on paravirtualization. In a BYOD environment, VMs can be implemented by using a special VM software designed for mobile devices. This way, separation of spaces is achieved by deploying different OSes on the mobile device and assigning one guest OS for each space.

In addition to separation of spaces, VMs can provide data protection by imple-

menting cryptographic primitives. However, they cannot meet the goals related to true space isolation, as the corporate data remains on the mobile phone. Further, they do not offer policy enforcement, they consume computational resources, and they can be intrusive to the user since additional software is needed.

For the case of paravirtualization, we can differentiate between heavy duty VMs and lightweight VMs. In the rest of this section we provide examples of heavy duty and lightweight mobile virtual machines.

3.1.1 Heavy Duty

According to Wang et al. [23], heavy-duty VMs allow the user to install multiple instances of an operating system. However, support from vendors might not be available. We present an example of a heavy duty VM as follows.

3.1.1.1 VMWare Horizon

VMware Horizon Mobile is an Android application that allows the user to run a guest OS on top of the main OS on the mobile device, creating a container for the corporate space, which is isolated from the personal space [33]

Barr et al. [29] describe the components of MVWare Mobile Virtualization Platform (MVP), which includes a VMM for the ARM architecture, an enterprise virtual phone and a remote device management software. The authors designed this solution considering the goals of portability, compatibility, security, low-complexity, performance, manageability and OEM time-to-market. In order to describe their virtualization platform, the researches discuss how some of the components of mobile devices are virtualized. For example, the VMM components and the VM images are encrypted and stored on the existing host file system.

In terms of security, VMWare MVP provides isolation by creating two separate environments. While the host environment corresponds to the user space and it does not offer any restriction, the enterprise environment is used for the corporate space and the administrators have control over the privileges in it. Further, applications in the user space cannot access the corporate space. Finally, while booting, a chain for trust is created in order to guarantee the integrity of the image corresponding to the corporate space.

One of the main drawbacks of this alternative is its limited scope of application, since only some devices from LG and Motorola are compatible [34]. Although this virtualization software is additionally provided with a management server software that allows the administrator to control the applications installed in corporate space, this type of solution is not present in iOS and it has been replaced by desktop virtualization technologies [35].

3.1.2 Lighweight

This type of virtual machine does not provide all the features that a heavy duty solution can offer as they do not allow the use of multiple guest OSes. However in terms of resources, they are not as demanding as the heavy duty alternatives. We present examples of lightweight VMs as follows.

3.1.2.1 Cells

Andrus et al. [28] present Cells, which is a virtualization architecture whose goal is to virtualize mobile phones. According to the authors, the current solutions do not provide an effective solution for virtualization that takes advantage of all the components that a hardware mobile device has. As Cells is a lightweight OS virtualization,

it only uses one OS and offers virtual namespaces that run multiple virtual phones.

There is a clever set of considerations that the researchers took into account. First, they exploit the fact that mobile OSes only show one application at the time. This way, only the virtual phone that is on the foreground is given access to the hardware. Second, in order to provide communication, each virtual phone is provided with VoIP capabilities, as opposed to use multiple SIM cards. Finally, in their implementation the authors of Cells note that their approach scales much better than heavy duty VM and it fully takes advantage of all the hardware that a mobile device offers.

Cells uses a unique kernel that is in charge of virtualizing identifiers, kernel interfaces, and hardware resources, which are then employed by the execution environment of each virtual phone. In order to provide isolated virtual phones, Cells uses namespaces, which are identifiers for the virtual phones. At the kernel level, Cells implements device driver wrappers, modifies the device subsystem and modifies the device driver. At the user level, namespaces are also used as a proxy mechanism in order to provide same features to devices that closed source. This mechanism is also used for settings that are particular to each user space. The file system is also virtualized using different mount namespaces, which allows different virtual phones to access the file system in an isolated fashion.

In order to manage the virtual phones, Cells includes a root namespace that is isolated from the other virtual phones, and which has access to the entire file system. During the initialization phase, a custom process is copied with the new namespace.

According to the authors, the advantages of Cells are in terms of scalability and security. For the former, the use of the same base system, the implementation of shared memory for virtual phones and the use of Android low memory increase the number of virtual phones that can be run. For the latter, the use of isolation techniques secure each virtual phone from the other.

Cells has been implemented and evaluated, being able to run up to five virtual phones in Nexus 1 and Nexus S without any significant overhead. Further, a human UI test showed that no visible performance degradation.

3.1.2.2 L4Android

Lange et al. [36] introduce L4Android, which is defined by its authors as a framework rather than a VM. However, the topic of their research can be classified into the VM category.

The authors of L4Android state that the security issues related to Android are inherent to its monolithic architecture and the permissions required by their subsystems. They state that if one of these subsystems has a bug, it might affect the rest of the components and increase all of its permissions.

L4Android focuses on component isolation, where the subsystems are encapsulated into components that run in protected domains. Communication between components is still possible. In order to provide security to the kernel, this solution uses L4Linux, a port of Linux to the researcher's microkernel. This microkernel is in charge of providing protection of domains, execution, communication, interrupt calls, scheduling, and the creation of objects and virtualization containers. Another component of L4Android is the runtime environment, which provides services and libraries to the applications. For the runtime environment, the researches selected L4RE. The VMM selected for this project is Karma, which provides paravirtualization, and supports Android as a guest OS.

Lange et al. [36] provide a set of evaluation scenarios, being one of them a BYOD environment, where there is a personal space and a corporate space. They mention that using L4Android it can be possible to securely unify both spaces in a single mobile device. In their example, two instances of L4Android were deployed.

The researchers mention that the isolation not only must be enforced between the guest OSes, but also at the user interface. Further, access to the hardware must be multiplexed by implementing drivers with multiple profiles but are shown to the guest OSes as a single device.

3.2 Agent-based Solutions

Agent-based solutions are alternatives that provide security policies and other features by installing a mobile application on the BYOD device. In the corporate network, there is another software that is in charge of establishing the set of rules and profiles that the different mobile devices will have. The mobile application needs special privileges on the mobile devices, as it is used to enforce configuration and profiles based on the rules determined by the administrators.

Although the most popular type of agent-based solution is Mobile Device Management (MDM), we can also find Mobile Application Managements (MAMs) and Mobile Information Management (MIM). In the rest of this section we will discuss these three alternatives.

Agent-based solutions are intrusive to the user, as the agent needs special privileges to monitor the mobile devices, as well as to enforce the security policies. These solutions do not provide any form of space isolation since the user space and the corporate space are stored on the mobile device. Agent-based alternatives are not resource-intensive and they might provide corporate data protection by enforcing the use of cryptography.

3.2.1 MDM

Leavitt [37], Eslahi et al. [38] and Scarfo [39] elaborate on MDM. In this solution, the main objective of the agent is to monitor the mobile device, transfer the application and user data to the server, and enforce policies and allow security mechanisms such as remote wiping. The agent is generally downloaded through application marketplaces, either public or owned by the enterprise. This way, the MDM server can execute commands on the BYOD device in order to lock down, control, encrypt, prevent the use of sensors, prevent information leakage and enforce policies. Additionally, MDMs allow the enterprise to distribute software on the mobile devices.

Wang et al. [23] list a series of drawbacks related to MDMs solutions. First, users can have more than one role in the enterprise and MDM have shown issues to handle that type of scenario. Second, enterprises can have policies that cannot be enforced by MDM alternatives, or have policies that are in direct conflict with the ones offered by the MDM software. Finally, since separation of spaces is not provided, it results impossible to implement different policies for the user data and applications, losing flexibility in the personal space. Also, Leavitt [37] points out that MDMs do not deal with hacker attacks, nor theft or loss of the mobile device.

In the market, we can find different MDM solutions, some of them named Enterprise Mobility Management (EEM). We provide a list of the most popular as follows.

3.2.1.1 AirWatch by VMware

AirWatch is a solution offered by VMWare for device management [40]. This solution allows managing devices in a central administration console that allows enrolling devices, and configuring and updating their settings. This alternative is presented either in the corporate network or as a cloud solution.

For mobile devices, this alternative is available for Android and iOS, among other platforms. In the enterprise side, it can be integrated with AD/LDAP directories to take advantage of domains, groups and device profiles, mobile applications and content. Device enrollment is available through an agent, a QR code, email or SMS. User authentication can be performed by username-password combination, directory services credential, token or proxy.

AirWatch offers tracking of unauthorized users, compromised devices and other risk. Additionally, the system notifies the administrators in case of detecting any risk. Further, administrators can send commands to the registered devices to either query for information or to take actions. These actions can be simple like sending messages or finding devices to more complex such as clearing passcodes, locking devices, performing a remote view or wiping the device.

3.2.1.2 Amtel MDM

Amtel [41] [42] [43] offers a cloud-based solution for both Android and iOS at the device side, while supporting LDAP/AD for the enterprise side. One of the features of is the fact that users can perform a self-enrollment in the system.

Amtel allows different type of actions that can be taken on mobile devices, such as monitoring, removing unauthorized devices, locking and removing passwords, wiping phone content, detection of rooted devices as well as blocking web content. For Android devices, GPS tracking is also available. Moreover, Amtel offers support for Samsung KNOX, a framework solution detailed in 3.6.2.2. In terms of device configuration, this solution offers settings configuration for email, VPN, Wi-Fi, Calendar, credentials and keys.

3.2.1.3 BlackBerry BES10

BlackBerry [44] offers a single point of control that integrates LDAP/AD for authentication and role-based administration. The devices supported by this solution include Android and iOS. Device activation is over-the-air and devices can be enrolled either by administrators using a combination of email and activation password, or by users.

This solution provides reporting tools to the administrators as well as configuration capabilities such as certificate distribution for accessing Wi-Fi, VPN or other corporate resources.

BES10 can enforce how devices connect to the network through Wi-Fi or VPN, disable the use of the camera, enforcing the password rules, detecting and rejecting rooted devices or devices that do not comply with the policies. Users can be notified about lack of compliance. Additionally, this solution provisions certificates. Actions that can be performed with BES10 include locking the device, password reset and wiping information from the device.

3.2.1.4 CA MDM

CA Technologies offers an MDM solution that includes a framework of policies for enrolling, configuring and onboarding devices [45]. Further, it offers analysis tools in order to track the devices, their expenses in terms of communication, and their security compliance.

The solution offered by CA Technologies can be implemented through the Cloud or installed inside the corporate network. On the device side, support for Android as well as iOS is provided.

Besides obtaining analytics from the devices, this MDM can also perform actions on them, such as locking, fully or selectively wiping the content of a device, remotely

locating a device, resetting its passcode, removing enterprise controls, and unregistering the device. On the server side, this solution can be integrated with LDAP/AD and certification authorities.

3.2.1.5 Citrix XenMobile Device Manager

This solution [46] [47] provides role-based management, configuration and security to mobile devices, as it can enroll and manage devices, detects if devices are rooted or out of compliance, as well as locate devices and monitor them. The actions that this product can perform range from blocking access to corporate data or locking the device to a full or selective wipe.

On the server side, XenMobile can be integrated with LDAP/AD to manage groups, users and profiles either as a cloud service or inside the corporate network. Moreover, it also provides support for certificates. Supported devices include Android and iOS.

In terms of administration, policies can be implemented based on the type of OS; and it allows to configure device properties such as passcodes, encryption, Wi-Fi and VPN. Enrollment can be performed via email or SMS through URLs or PINs.

3.2.1.6 Dell EMM

This alternative from Dell [48] provides support for both Android and iOS devices. Since this solution is offered on the Cloud, no requirements are needed for the enterprise other than a web browser.

In terms of device management, Dell EEM offers over-the-air configuration, with policies automatically updated from the Cloud. Enrolling and registration is performed directly by the users. Further, different policies can be implemented using

a graphical interface with geo-location. This solution provides real time of device tracking, its applications and its configuration.

Finally, the type of actions that can be performed include simple tasks such as querying the device, auditing it or sending it a message, to more complex one such as locking the mobile device, clearing its passcode, unregistering it or even wiping the content in it.

3.2.1.7 Good MDM

The MDM alternative offered by Good Technology [49] [50] supports a cloud-based central location to configure and control any device over the air. On the server side, this solution can be integrated with Microsoft AD, allowing group management. On the user end, Android and iOS are supported by this MDM.

This alternative can enforce policies such as password, device encryption, camera, Wi-Fi, and VPN configuration. Other features offered are root detection, password reset, and reports such as device status, installed apps, utilization and expenses. Enrollment can be performed by the user using URLs or through a mobile application. In addition, actions such as data wiping and configuration removal are also provided.

3.2.1.8 IBM MaaS360

This cloud-based solution by IBM [51] [52] offers support for both Android and iOS mobile devices, while integrating LDAP/AD run by the enterprise. This allows creating customized policies and applying them to users and groups.

MaaS360 provides configuration for email, calendars, contacts, Wi-Fi and VPN. It allows to authorize or reject new devices in the network, as well as actions such as locating lost or stolen devices, resetting passwords, sending messages to devices, updating configuration settings, and deleting the corporate data and MDM config-

uration in the device. Further, this solution enforces security by specifying policies related to passcodes and encryption, as well as detecting rooted devices and creating location-related policies.

In terms of monitoring, this solution offers features that include reports of hardware and software inventory and mobile expense management, configuration and vulnerability. Finally, enrollment can be performed using SMS, email or URL.

3.2.1.9 McAfee EMM

EEM [53] is a broader solution that offers MDM capabilities. It can support both Android as well as iOS devices. Management of the devices is performed through a centralized infrastructure located inside the corporate network, which supports LDAP/AD and groups.

The main features of this solution are enforcing the use of anti-malware, detection of rooted devices, enforcement of authentication, authorization and encryption, access restriction based on device type or OS. Additionally, it enforces policies related to password, restriction related to download applications, access to the camera, selecting which applications can open attached files, or backing up information on the Cloud. This solution can deliver configuration provisioning for VPN and Wi-Fi. Finally, it can remotely lock devices, wipe the entire device or just remove corporate information on devices that have been lost or stolen.

In terms of monitoring and report, this alternative can provide reports related to mobile policy compliance, threat events, application reputation, and other metrics. Moreover, when a user is out of compliance, he or she receives automatic notifications explaining the reasons why the device is in this situation.

3.2.1.10 Microsoft Intune

Intune is a cloud-based MDM solution by Microsoft [54] [55], meaning that the enterprise does not need to deploy any additional infrastructure in the network. On the user side, devices running Android as well as iOS are supported. Intune is part of Microsoft's suit of solutions on the Cloud, which includes user administration through Azure Active Directory Premium. Configuration is also handled by Intune, as it is able to provide certificates for email, Wi-Fi and VPN.

Users can perform the enrollment process by accessing a web page. Intune can be used for implementing of policies in order to reject access to the corporate resources from unauthorized or out of compliance devices. Further, this MDM allows actions such as password reset, device lock, rooted device detection, enforcement of data encryption, and full wipe for lost or stolen mobile devices.

3.2.1.11 MobileIron EMM

This solution from MobileIron [56] [57] [58] can be implemented either by deploying the infrastructure inside the corporate network or by using a service hosted on the Cloud. On the other end, support is provided for Android and iOS.

Mobile Iron EEM offers integration with LDAP/AD, including role-based access and implements group policies, configuration provisioning for Wi-Fi, email and VPN is given over-the-air. This solution additionally offers certificate distribution, as well as rooted device detection. Further, this alternative provides email attachment protection. Actions that can be performed on the devices include recovery of lost devices, selective wipe, broadcast messages and notification of current compliance. Finally, for reporting purposes, MibileIron collects data related to device, applications, user and status, which are displayed in a management and reporting console.

3.2.1.12 SAP Afaria

SAP Afaria [59] [60] is a solution that offers MDM capabilities for Android and iOS mobile devices. This solution can be implemented on the Cloud, as well as on-premise. Enrollment can be performed by the users. SAP Afaria includes group policy.

Reporting, monitoring and managing capabilities are performed in a centralized console. The console shows data related to device tracking, device type, usage monitoring and logging for applications and usage statistics. Additionally, automatic alerts and controls can be implemented to provide real-time cost management.

For configuration provisioning, SAP Afaria offers certificate delivery for single sign-on to the corporate resources and data encryption, as well as email, Wi-Fi and VPN configuration. This solution can enforce password policies, peripheral controls such as locking down the camera. Activities performed on the mobile device include remote lock and data wiping, or rejection of devices that do not comply with the corporate policies.

3.2.1.13 SOTI MobiControl

SOTI offers a MDM solution [61] that can be deployed in the corporate network or as a service based on the Cloud. This alternative can be integrated with LDAP, allowing the use of group configuration. The list of devices supported includes Android and iOS. Mobi Control provides a self-service portal where users can reset password, wipe data, lock devices or find lost devices through geo-location information.

MobiControl offers the administrators monitoring as it can provide statistics on status of connectivity, device information, location and active policies, as well as monitor device, custom data, and server-specific incidents. In addition, this solution can determine when a device has been rooted. This way administrators can determine

when a device has been compromised and take actions. Administrators can additionally enforce policies over incoming and outgoing calls. Finally, policy enforcement includes strong passwords and the use of cryptographic primitives.

3.2.1.14 Symantec Mobile Management

Symantec's solution for MDM [62] is a centralized approach that can be developed inside the corporate network or on the Cloud. It implements LDAP/AD support for authentication and group membership and role-based access control. The devices supported by this solution include Android and iOS.

Synmatc's product offers enrollment to the users as a self-service process. This solution provides automated configuration and certificates that facilitates network access for email, VPN and Wi-Fi. In terms of policies, it allows password enforcement, application restrictions and remote wipe. Additionally it detects when devices have been rooted.

Administrators can obtain custom or predefined reports including information related to user, devices, applications, and profiles.

3.2.1.15 Comparison

In Table 3.1, we provide a qualitative comparison between all the MDM solutions that we have listed. It can be seen that all the solutions described provide similar features in term of the configuration provisioning, the actions that can be take on the mobile devices, and the type of supported mobile OSes. However, there are differences in terms of how the solutions are deployed, since some of the alternatives are only cloud-based solutions, and the possibility of provision geo-located data, which let the enterprise know where the mobile device is and enforce policies based on that location.

Solution	Features											
	On-premise	Cloud Implementation	LDAP/AD Integration	User Self-Enrollment	Group Administration	Android/iOS Support	Geo-Location	Configuration Provisioning	Root Device Detection	Data Wiping	Password Reset	Messages to Users
AirWatch by VMware [40]	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
Amtel MDM [41] [42] [43]		✓	✓	✓		✓	✓	✓	✓	✓	✓	
BlackBerry BES10 [44]	✓		✓	✓		✓		✓	✓	✓	✓	✓
CA MDM [45]	✓	✓	✓			✓	✓	✓		✓	✓	
Citrix XenMobile Device Management [46] [47]	✓	✓	✓		✓	✓		✓	✓	✓		
Dell EMM [48]		✓		✓		✓	✓	✓		✓	✓	✓
Good MDM [49] [50]		✓	✓	✓	✓	✓		✓	✓	✓	✓	
IBM MaaS360 [51] [52]		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
McAfee EMM [53]	✓		✓		✓	✓		✓	✓	✓		✓
Microsoft Intune [54] [55]		✓	✓	✓	✓	✓		✓	✓	✓	✓	
MobileIron EMM [56] [57] [58]	✓	✓	✓		✓	✓	✓	✓	✓	✓		✓
SAP Afaria [59] [60]	✓	✓			✓	✓		✓		✓		
SOTI MobiControl [61]	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
Symantec Mobile Management [62]	✓	✓	✓	✓	✓	✓		✓	✓	✓		

Table 3.1: Comparison between the different types of MDM solutions.

3.2.2 MAM

After reviewing MDM solutions, we will focus on the second category of agent-based solutions: MAM. According to Leavitt [37], Eslahi et al. [38] and Scarfo [39], MAM goal is to manage and limit the mobile applications on the BYOD devices. Additionally, MAMs control that only authorized applications can access corporate data. This solution allows the enterprise to implement policies on the use of mobile applications by rejecting unauthorized applications, distribute them to devices, removing them,

updating them, creating backups of them, auditing them, as well as enforcing policies related to the behavior of the mobile applications. This way, only the applications that are white-listed can access the corporate resources. Further, applications not related to the enterprise get isolated from the corporate resources. However, communication between applications can be negatively impacted, and no data protection is provided.

One important difference between MDM and MAM is that the former performs its control at the hardware layer, while the latter control the software on the device. Consequently, corporate policies could be enforced only to the applications related to the enterprise. MAM and MDM are complementary solutions; this is the reason why MAM solutions are integrated into MDM products.

3.2.3 MIM

Finally, after describing MDM and MAM, we review MIM solutions. According to Eslahi et al. [38] and Scarfo [39], MIM alternatives focus on providing a centralized entity that stores all the corporate data, as opposed to leting the mobile devices store such content. Since all the corporate information is stored in a single place, it is easier to guarantee the security of the information. One of the main advantages of this solution is that it provides synchronization across different devices. Additionally, this service can be hosted on the Cloud, with the advantages and disadvantages presented in the following section.

3.3 Cloud-based Solutions

Leavitt [37] provides the use of solutions that rely on cloud storage in order to offer mobile access to data as well as applications. These solutions provide space isolation

since the personal data can reside on the mobile device, while the corporate data is stored on the Cloud (personal data can also be stored online under a different repository). True space isolation is also guaranteed, because the corporate data is not hosted on the mobile device. However, Leavitt [37] points out that isolation is no longer guaranteed once the data gets downloaded to the mobile device. Further, access to the corporate data must be properly configured and managed by either the user or a system administrator.

These solutions additionally provide protection to the corporate data, as cloud-based solutions offer encryption. Cloud-based solutions are not intrusive to the user, since there is no need to install any special software to access the online data through a default web browser. However, many cloud-based storage providers offer mobile applications that directly access the data repository, without browsing the web. Finally, these solutions do not demand a high amount of resources from the mobile device, other than network connectivity.

For most of the enterprises it is impossible to deploy a cloud-based solution that satisfies their needs. Consequently, enterprises turn to third-party solutions that address these needs in the form of cloud services for a fee. However, from the security perspective, relying on a third-party implies losing a certain degree of control over the corporate data and the need of trusting on the cloud storage service provider. For example, IBM has banned the use of cloud storage and other cloud services as its employees were storing confidential data outside IBM's network [24]. Additionally, cases of mismanagement by the cloud storage provider have been reported [63]. Further, because sensitive data is stored on cloud services, their service providers have become the target of attacks [64] [65]. Finally, the case of password re-use must be taken into account, since many users use the same type of credential for different services, a repository that contains work-related data can be compromised because the log-in

credentials userd for other services have been exposed [66] [67]. The enterprise must trust on the provider with respect of the security measures the latter implements as well as how the service is administrated.

There are two main categories in solutions for cloud storage, one that targets small and big business, and another for consumers. In the first group we can find solutions such as Acronis True Image Online [68], ADrive [69], Bitcasa [70], Carbonite [71], Copy (owned by Barracuda Networks, Inc) [72], CrashPlan [73], Egnyte [74], Hightail [75], IDrive [76], JustCloud [77], Livedrive [78], Mankayia [79], Mozy (owned by EMC Corporation) [80], OpenDrive [81], SOS Online Backup [82], SpiderOak [83], SugarSync [84] and Zip Cloud [85]. All of them provide mobile applications for both iOS and Android, making them suitable for BYOD environments. Additionally, it can be seen that there are many companies in this area; however in a report from Ovum cited by Business Cloud News [86] it is claimed that 89% of employees use consumer-focused cloud storage solution instead of a business-focused alternative. Further, the consumerization of these services lead some these business-oriented solutions to also compete in the end-user market.

There are fewer options available in the end-user market. However, these solutions additionally offer service plans for business use. A list of the most popular consumer-oriented solutions is presented as follows.

3.3.1 Box

Box [87] is solution that only focuses on cloud storage. It offers several plans ranging from 10GB to unlimited storage, with different fees for different storage capacities. These plans are divided into personal, starter, business, enterprise and elite. This service provides support for mobile devices, since it provides mobile applications for

Android as well as iOS. From the security perspective, Box offers SSL and At Rest Encryption with 256-bit AES, and Two-factor authentication. However, mobile applications for end users do not offer Two-factor authentication but a simple password protection, as multi-factor authentication and password enforcement is only available for the enterprise and elite plans. In terms of content access management, the free alternatives do not provide user management, while the business, enterprise and elite plans offer this feature. Additionally, these last three services also offer MDM integration. Privacy is also addressed, as Box has been certified for EU and Swiss Safe Harbor frameworks and it is in compliance with HIPAA regulations. Finally, Box provides the possibility of file edition through integration with third-party vendors.

3.3.2 Dropbox

Dropbox [88], just like Box, is a service that only focuses on cloud storage. The plans offered by this solution are classified into basic, pro and business, with storage capacities that range from 2GB for a single personal use, to unlimited for the business plan. Mobile devices are supported by this solution, with mobile applications for Android as well as iOS. Dropbox provides transfer security through SSL and storage encryption by implementing AES-256. This solution offers Two-Factor authentication but only when a mobile device is linked to an account for the first time. Additionally, password protection can be enabled for the mobile clients. Dropbox supports group sharing, meaning that they implement a role based access management on the folders and files to be shared, but this feature is only available for the business plan. Mobile file editing is not natively supported, but is available through the use of third-party software. Finally, in terms of standards and regulations, Dropbox complies with ISO 27001, SOC 3 for Security, Confidentiality, and Integrity, SOC 2 for Security,

Confidentiality, Integrity, and Availability, SOC 1 / SSAE 16 / ISAE 3402 (formerly SAS 70), PCI DSS, and U.S.-E.U. and U.S.-Swiss Safe Harbor.

3.3.3 Google Drive

Google Drive [89] is the cloud storage service by Google. Just like any other solution in the market, they offer plans for personal use and for business. The former offers 15GB, which are shared by all the services offered by Google, while the latter offers unlimited storage capacity. Drive provides integration with mobile devices, as mobile applications for Android and iOS are available. When it comes to security for data transfer and storage, Drive claims that files are encrypted during data transfer from the device to Google, between Google's datacenters and when it is stored on the mobile devices. However, it is not mentioned which standards are implemented to achieve these features, nor the fact that the stored files are encrypted in Google's servers. Google offers Two-step verification for their accounts, but in mobile devices this feature is only used when the device is verified for the first time. For access management, Drive offers data sharing and group sharing. The latter is achieved by setting up a group email. Since Google offers also Google Docs, edition of files is available without the need of a third-party application, although third-party support is available. Drive has been certified with SSAE 16 / ISAE 3402 Type II, SOC 2 and ISO 27001. Additionally, for privacy, Drive complies with FISMA, FERPA, and HIPAA, and adheres to the Safe Harbor Privacy Principles.

3.3.4 iCloud

iCloud [90] is the storage solution offered by Apple Inc. This service is only available for end-users, with storage capacities that go from 5GB to 1TB. Just like any software

from Apple, iCloud is tightly integrated with the rest of the software suite that the company offers, such as Notes, Calendar, Mail, Photos, etc. For BYOD access, the service provider only offers mobile applications to devices that run iOS, leaving Android support for third-party applications. File editing is offered through Pages, Sheets and Slides. Since this solution is not offered to business, file sharing is not an implemented feature. In terms of encryption, Apple claims that all data in transit is encrypted using SSL, while data stored is encrypted using a minimum of AES-128. Two-step authentication is also implemented for mobile devices, but it only employed the first time a device is used to access the service.

3.3.5 OneDrive

OneDrive [91] is the cloud storage service provided by Microsoft. This solution provides end-user plans and business plans, with storage capacity from 15GB to 1TB. OneDrive is available for mobile devices through mobile applications for Android and iOS. Document editing in mobile devices is supported by OneDrive with an Office356 subscription, otherwise the user must use a third-party application. In terms of security, OneDrive implements SSL/TLS for file transfer, but it does not encrypts the files when they are stored in Microsoft's cloud servers [92]. Access management on stored data has the feature of group sharing. OneDrive implements Two-factor authentication that is also available for their mobile applications. Microsoft's cloud infrastructure has been certified for ISO 27001:2005, PCI DSS, FredRAMP P-ATO, FISMA, and it is in compliance with SSAE16/ISAE 3402 SOC 1, AT101 SOC 2 and 3, and HIPPA.

Table 3.2 provides a summary and comparison of the features for different consumer-oriented cloud-based solutions. We can observe that generally all the solutions provide

Solution	Feature								
	Business Option	Unlimited Storage	BYOD Support	Document Editing	Secure Transfer	Secure Storage	Two-Factor Authentication	Group Management	Privacy compliance
Box [87]	✓	✓	✓	✓		✓	✓	✓	✓
Dropbox [88]	✓	✓	✓	✓		✓		✓	✓
Google Drive [89]	✓	✓	✓	✓	✓	✓		✓	✓
iCloud [90]			✓	✓	✓	✓			
OneDrive [91]	✓		✓	✓	✓		✓	✓	✓

Table 3.2: Comparison between different consumer-oriented cloud-based solutions.

the same features, with the exception of iCloud, that does not provide the features proper of an enterprise-focused solution. Additionally, we can mention that only Box and OneDrive provide support for Two-factor authentication in their mobile applications.

So far we have discussed cloud-based solutions that are closed-source, meaning that users and enterprises must trust what the providers claim. For example, a provider must implement a security feature such as data encryption, but does do not ensure that the implementation is correct and free of security issues. Open-source cloud-based storage solutions try to overcome these concerns by publishing the code they use to implement their solutions. In this space we can find different alternatives such as Seafile [93], ownCloud [94], git-annex [95], SparkleShare [96], Syncthing [97], Stacksync [98], OpenStack [99]. On the one hand, the enterprise can deploy and maintain a server to provide this service to their user, while some of these alternatives offer paid versions that offer more features and technical support. On the other hand, most of them do not offer an iOS mobile application, which limits their capacity to compete in the consumer market.

3.4 Virtual Private Network (VPN)

The use of VPNs for BYOD environments is presented by Wang et al. [23], Leavitt [37], Hayes [8] and Leong [100]. According to Peterson et. al. [101], a VPN is a logical network that is deployed on top of a physical network, with the purpose of interconnecting different nodes from different networks as if they were part of the same private network. For example, for BYOD environments, the most common case is presented when a mobile device that is connected to the Internet through a Wi-Fi access point needs access to information located inside of the corporate network. In this scenario, we say that a tunnel is created between the mobile device and the network. The concept of tunneling is straightforward; it means to encapsulate one packet inside of another. In the case of a VPN, the mobile device encapsulates the packet for the private network in a packet for the Internet. This way, the Internet forwards the Internet packet normally, but when the Internet packet reaches the border of the corporate network it is opened and then the packet for the private network is forwarded accordingly inside of the corporate network. Therefore, the mobile device can join the corporate private network and access the data.

Tunnels can offer security with the addition of encryption techniques, as the encapsulated packet can be encrypted. This provides confidentiality to the communication, and it can also offer integrity of the messages. However, the use of VPN creates a computational cost, as the information must be encrypted, and decreases the throughput of the message, since we need to increase the size of the overhead by sending additional information.

As mentioned before, VPNs only protect the communication between the BYOD device and the corporate network. Consequently, it can be argued that this technology only covers part of the goal of corporate data protection, since once the data is

retrieved to the mobile device, it can no longer be secured. For example, if the mobile phone is lost or stolen, all the corporate data on the phone is at risk. VPNs do not provide any type of space isolation nor true space isolation, as nothing prevents the personal and corporate data to be mixed on the mobile device. Additionally, VPNs do not offer any type of security policy enforcement. However, this type of solution is not resource intensive nor intrusive to the user.

Leavitt [37] states that although some enterprises have implemented the use of VPNs and have set security policies to enforce the use of them, this does not prevent the user from accessing insecure networks such as hotel Wi-Fi, where the mobile devices can become targets of attacks. In this fashion, if a mobile device gets compromised, the corporate network can become compromised even if VPN was used to access it.

Solutions in the space of VPN include standards and commercial applications. In the case of standards, we have included the ones that are supported by Android and iOS, since they are the most popular mobile operating systems. We present these solutions as follows.

3.4.1 Standards

Standards are a series of protocols open to the public and well documented, generally through RFCs.

3.4.1.1 L2TP/IPSec

L2TP/IPSec is a combination of two protocols. First, Layer Two Tunneling Protocol (L2TP) [102] enables the use of VPN between the mobile device and the corporate network by offering a dynamic mechanism for tunneling layer two circuits over a

packet-oriented network, by specifying how to create, maintain and remove these circuits. But this protocol does not provide any type of security features. Second, the Internet Protocol Security (IPSec) [103] is a layer-three protocol that provides the security features of encryption and integrity. In order to achieve them, IPSec relies on four main elements: Security Protocols, Security Associations (SA), Key Management and Cryptographic algorithms. The security protocols can be the Authentication Header (AH) protocol, which provides only integrity, or Encapsulating Security Payload (ESP), which provides both encryption as well as integrity protection. An SA is a unidirectional connection that supports security services, identified by a number called Security Parameter Index (SPI). Key management is provided by the Internet Key Exchange (IKE) protocol [104], which is in charge of automatically creating the keys needed for the SAs. Finally, the cryptographic algorithms used by this protocol depend on the type of the security service that is needed [105]. For authenticated encryption AES-GCM with a 16 octet ICV should be supported and AES-CCM may be supported. For encryption only, NULL and AES-CBC must be supported, AES-CTR and TripleDES-CBC may be supported, and DES-CBC must not be supported. For authentication only, HMAC-SHA1-96 must be supported, AES-GMAC with AES-128 and AES-XCBC-MAC-96 should be supported, and NULL may be supported. The interaction between L2TP and IPSec is described in Patel et al. [106].

3.4.1.2 PPTP

The Point-to-Point Tunneling Protocol (PPTP) [107] was developed by a vendor consortium, and it describes a new alternative to route information through point-to-point protocol (PPP) [108]. PPTP defines a client-server architecture in order to decouple the tasks of a Network Access Server (NAS), while at the same time offering VPN capabilities. In this protocol, the server is called PPTP Network Server

(PNS), and the client is referred as PPTP Access Concentrator (PAC). In the PPTP scenario, two communications are used at the same time, a Control Connection that establishes, manages and releases the tunnel session, and the tunnel between the PAC and the PNS. This tunnel transports Generic Routing Encapsulation (GRE) [109] encapsulated PPP packets. Security features in PPTP can be achieved by implementing Microsoft Point-To-Point Encryption (MPPE) Protocol [110], which implements encryption by using RSA RC4. However, the authentication protocol used by Microsoft MS-CHAPv1 and MS-CHAPv2 presents security issues and its use is discouraged [111].

3.4.1.3 Cisco IPSec

Cisco IPSec [112] is the implementation of IPSec by the vendor, with the same features in term of encryption and authentication by using AH and ESP, as well as a key exchange protocol using IKE, and the implementation of SAs to describe the connections. Additionally, there is a certificate management by employing Simple Certificate Enrollment Protocol (SCEP). The cryptographic primitives used by this protocol are Diffie-Hellman in the IKE protocol, DES for packet encryption and MD5/SHA for data integrity. The difference between Cisco IPSec and L2TP/IPSec is that in the former no Layer 2 protocol is predefined.

3.4.2 Commercial Applications

Vendors provide their own VPN solutions. These alternatives are presented in the form of mobile applications for both Android and iOS. On the enterprise side, compatible network hardware must be installed to offer access to these platforms.

3.4.2.1 Cisco AnyConnect

AnyConnect from Cisco also provides secure VPN access by using Datagram Transport Layer Security (DTLS), IPsec (IKEv2), and TLS (HTTP over TLS/SSL) [113]. In order to provide encryption, the cryptographic primitives implemented in these protocols are AES-256 and 3DES-168 as well as NSA Suite B algorithms, ESPv3 with IKEv2, 4096-bit RSA keys, Diffie-Hellman group 24, and enhanced SHA2. For authentication, AnyConnect is compatible with RADIUS, RSA SecurID, Active Directory/Kerberos and LDAP. It also offers the possibility of multi-factor authentication by combining certificates and usernames with passwords. In addition, this solution offers per-app VPN functions for iOS.

3.4.2.2 Juniper Junos Pulse

This product provides remote user to the corporate network by implementing SSL VPN access. According to Juniper [114], Junos Pulse provides secure and authenticated access for authorized users by using a dual-transport full layer three VPN connectivity with granular access control. This dual-transport feature is achieved with the implementation of SSL and ESP. Additionally, this solution has the feature of endpoint integrity assessment, which checks the user device prior to authentication, based on policies. Junos Pulse can be integrated with MDM solutions, leveraging the management of the BYOD devices.

3.5 Trusted Environments

Trusted environments are solutions that deal with the fact that mobile devices might run applications that cannot be trusted with sensitive information. There are different alternatives to deal with this situation as some of the solutions are software-based

and focus on warning the user, while others are implemented in the hardware and address the problem by creating isolated environments.

3.5.1 Trusted Execution Environment (TEE)

The use of TEEs is discussed by Ekberg et al. [115]. In TEEs a secure processing environment is separated and isolated from the processing environment that normally is used by the OS and the applications, which is defined as Rich Execution Environment (REE). This way, applications are provided with better security features by dividing them into two parts. The first part does not contain any sensitive operations and run in the REE, while the section that handles the sensitive operations run in the TEE. According to the authors, mobile devices that include TEEs have the potential of replacing any type of hardware used as an access token (i.e. keyfobs for two-factor authentication, RF IDs). TEE is presented in most of the mobile devices, as the vast majority of them include ARM's TrustZone [32].

TEEs can be combined with a Trusted Platform Module (TPM) technology for mobile to increase the security features [116]. TPM [117] implements security primitives in a separated cryptographic co-processor in order to enable trust computing platforms. The cryptographic operations supported by the co-processor are Asymmetric key generation (RSA), Asymmetric encryption and decryption (RSA), Hashing (SHA-1), and Random number generation (RNG). Moreover, the mobile version not only implements all these features but also includes secure boot, the introduction of an alternative implementation such as a firmware TPM, and support for several parallel TPM instances in the same device.

If we analyze the goals for a secure BYOD environment, we find that TEE can offer space isolation as the data applications related to the enterprise can be separated

from the personal data, it is not intrusive to the user and it does not require extra resources from the device since it is built in the smartphone. However, it does not provide any data encryption and the corporate data cannot be protected. Further, it does not provide any means to implement security policies from the enterprise. Finally, despite of the fact that space isolation is provided, no true space isolation is present since the corporate data still resides at the mobile device.

3.5.2 TrustDroid

Zhao and Osono [118] introduced TrustDroid. In their research, the authors mention the shift in the paradigm from trusted nodes in a trusted network to the combination of a cloud environment and a BYOD environment, where data and applications reside on the Cloud and user's mobile devices are not considered trustworthy. Following their argument, corporate data and applications cannot be protected.

TrustDroid addresses the problems related to this scenario by performing a static analysis based on taint tracking. It assumes that the BYOD device runs applications that are not trusted by the enterprise, which can lead to data leakage. TrustDroid detects such scenarios and warns the user about this situation.

One of the features of this solution is that it can work in two modes, an off-line mode where the analysis is performed by the corporate resources, and an on-line mode where the analysis is performed by the mobile device, which creates an overhead of resource consumption. However, TrustDroid provides means to set the granularity of the analysis, reducing the recourse consumption overhead.

TrustDroid works by analyzing the bytecode of applications and tries to find entries that manipulate sensitive data, according to a set of predefined rules. Sensitive data is marked with a tag. Then, this tag follows the sensitive data as it processed

by the bytecode.

If sensitive data is found, then it is marked with a tag that follows the data as the latter flows through the byte code. Lastly, if the tagged data reaches a sink, such as a network interface, the operation is flagged as leak activity by the byte code.

TrustDroid creates a trusted environment by notifying the user which applications might leak sensitive information. On the one hand, it is not resource intensive since the analysis has different degrees of granularity. On the other hand, this solution does not provide any separation of spaces, it does not implement security policies, and it can be intrusive to the user since its implementation is an application that might analyze personal applications. Finally, even if data leakage can be prevented, it does not fully provide corporate data protection since it does not address the case where the data resides on the mobile device and the latter becomes lost or stolen.

3.5.3 MOSES

Russello et al. [119] are the authors of MOde-of-uses SEparation in Smartphones (MOSES), a policy-based framework for enforcing software and data isolation on the Android platform, using a lightweight approach.

These researchers focus on virtualization by using a controlled software isolation. Security profiles are then applied to contexts that determine when a profile can be activated. These contexts are characterized by a combination of low level features such as time and location, and high level features like reputation and trust level. MOSES provides a GUI that users can employ to specify the profiles and contexts. Further, profiles have a high level of granularity by allowing controling single objects and applications. Finally, the switching between profiles can be either manual or automatic.

MOSES offers a separation of data and applications in contexts that are isolated from each other in a single mobile device. This way, the user space can be separated from the corporate space. Compartments called security profiles are deployed in order to achieve this isolation. These profiles are a group of policies that manage the privileges of applications and the access to data. Since this solution allows the management from an external administrator, it presents a similar characteristics to the ones found in MDM solutions.

According to their work, there are two versions of MOSES. The first one, and oldest, is based on TaintDroid [120] which labels data in order to know how sensitive information is propagated by mobile applications on the mobile device. However, the latest version of MOSES is based on a lightweight file system virtualization architecture implemented at the Linux kernel, in order to achieve separation of spaces.

MOSES is a framework and includes several components. The first one is the ContextDetectorSystem, which is in charge of detecting when contexts are activated or deactivated. When one of these activities happen, the ContextDetectorSystem sends a message to the SecurityProfileManager, which is in charge of establish the security profiles for contexts. After a security profile is activated, the SecurityProfileManager informs MosesHypervisor. The latter is the central security authority, which handles the access to resources. Then MosesHypervisor delegates the policy checking to the MosesAppManager and the MosesRulesManager, which decide which applications can be run in the context and manage special rules, respectively.

The MosesPolicyManager is in charge of controlling the security policies by creating, modifying and deleting them. Further, it allows the user to manage contexts. MosesTaintManager is responsible of managing the database that contains the taint values used by the system. In order to enforce separated security profiles MOSES uses MosesReaper to detect and stop processes that are no longer required. Moses-

Mounter performs data isolation by using directory polyinstantiation [49], which allows to mount a directory as different instances based on some parameters. Finally, in order to change between security profiles, MOSES offers MosesSpChanger and MosesPolicyGui.

The researchers have evaluated MOSES using a Google Nexus S phone, an Android device. They did not find significant overhead for battery consumption and storage. In terms of overhead produced by security enforcement, MOSES is much less efficient compared to TaintDroid. For microbenchmarking, MOSES has an inferior performance than a stock Android, but a comparable performance to TaintDroid.

The authors of MOSES have noticed a series of deficiencies in their framework. For example, if an application obtains root access, then it can bypass MOSES security enforcement. Additionally, this architecture does not provide isolation for SD cards. Further, MOSES has only been developed for Android platforms, leaving iOS without an implementation.

MOSES offers space isolation, data protection and policy enforcement by design. However, it does not provide true isolation, as the corporate space is still located on the mobile phone. Further, this solution is resource intensive as it implements paravirtualization on the mobile devices, and it results intrusive to the user as modified Android software must be installed on the BYOD device.

3.6 Framework

The solutions presented so far focus on addressing specific issues related to the challenges in BYOD environments. Frameworks, however, are composed by a set of the previous solutions. The final purpose of a framework is to provide a global and integrated solution that covers all the goals for a secure BYOD environment. Frameworks

are alternatives that present a complex architecture, as they have components on both the BYOD device as well as at the corporate network. We can classify frameworks based on their origin in research-based or commercial alternatives.

3.6.1 Research-based Solutions

These solutions are found in the research community. Generally, they are not developed by any commercial company. We describe research-based solutions as follows.

3.6.1.1 2-Tier Access Control (2TAC)

Chung et al. [121] propose 2TAC, a distributed access control architecture for BYOD. The goal of this architecture is to solve the trust issues related to BYOD devices that access critical information in an enterprise. The researchers identify two main roadblocks that must be addressed in order to have a secure mobile device and an appropriate access control: (1) new malware and sophisticated attacks are released constantly, and (2) mobile devices are pieces of hardware with limited battery life. According to the authors, 2TAC is a framework that uses different concepts such as social networking, tagging system, malware-scanning using the Cloud, Role-Based Access Control (RBAC) systems that leverage the Cloud and virtualized security profile for mobile devices. The architecture of 2TAC consists on the Device Control Tier located at the mobile device, and the Cloud Control Tier located on the Cloud.

First, the Device Control Tier includes a profile virtualization that allows the device only to run certain tasks in function of the location and the time by specifying permissions, settings, applications, resources, as well as how often the malware-scanning is performed; and a light anti-malware software which only scans fundamental parts of the device, such as the kernel or the file system. This anti-malware can be also

started by the user or the administrator at any point.

Second, the Cloud Control tier performs all the management capabilities and it contains a profile management system, multiple anti-malware software scanners, access to user's logs and a trust management system. The profile management system offers the possibility of create, modify and delete multiple profiles to be used on the BYOD devices. Once a mobile device tries to access data, two tests are performed: first the device is analyzed to know if it has the proper profile and then the device is checked against the trust management system. The anti-malware software at the Cloud is more complex than the one on the mobile device, which let the system perform extensive analysis without considering the consumption of resources. Just like the light anti-malware, the cloud version can be run on-demand or on a schedule basis. In order to perform the analysis on the Cloud, a snapshot of the processes, applications and data is sent from the mobile device to the Cloud. The access record component is a set of logs containing information such as when, where and how devices accessed the corporate data. These logs can be used to determine when a mobile device has been lost or stolen based on the location of the device. Finally, the trust management system is in charge of data access and social network by implementing the use of tags on devices and corporate data. Tags contain information related to the user that modified a set of data, as well as how and when it was modified. According to the users, the implementation of this system is to eliminate the threat of a single point of failure by implementing this task in a distributed fashion. The social aspect of this implementation is achieved by introducing tags that are associated with different departments in the enterprise.

When it comes to security policies, 2TAC provides a flexible policy management by defining the type of access based on the enterprise's needs. However, the authors suggest four straightforward types of profiles that can be implemented using this

framework: untrusted user, basic user, advanced user, and super user. The difference between these types of profiles is the knowledge on security they demonstrate and how often the security scanning must be performed on their devices, and on the type of access they are granted through the corporate network.

According to their publication, the authors state that an implementation of 2TAC is planned as future work. This implementation will use Android for the mobile devices and Amazon Web Services as the cloud provider.

2TAC clearly provides policy enforcement and it is not resource-intensive to the user, as most of the scans are performed on the Cloud. Additionally, some degree of separation of spaces is offered through the use of profiles. Despite of this, 2TAC does not implement true space isolation because both the personal data and the corporate data reside at the mobile device all the time. Further, no cryptographic algorithms are used in order to maintain the corporate information protected. Finally, the fact that the device does not have space for the personal space means that the data and the applications of the user can be analyzed by the enterprise.

3.6.1.2 BYOD Security Framework (BSF)

Wang et al. [23] present another security framework for BYOD environments. Due to the fact that RMS is based on this framework, we provide a full description in Section 4.1.

3.6.1.3 Security Service Architecture (SSA)

Titze et al. [122] address the problem of mobile applications that do not comply with the corporate security policies. According to the researchers, marketplaces such as Google Play provide mechanisms to detect malware before they are downloaded and installed. However, in BYOD environments relying on the mobile application

marketplace can lead to security risks, since the enterprise cannot determine which requirements have been used by the marketplace to authorize or reject the mobile application. Moreover, even if these requirements are known, the enterprise must be able to define which requirements are needed according to its needs, instead of fully relying on the requirements decided by a third-party.

First, one of the main claims of the authors address the issues related to Google Play and Apple's App Store. In the former, the applications are uploaded, then analyzed and finally they are available to the users. However Google does not detail how malware is detected. The latter has a review process but no information on how this process works. Additionally, both marketplaces only have access to the binary files, without analyzing the source code of the mobile applications.

Second, the authors describe the techniques used to detect malware in mobile applications, being Android the most tested platform. However, they state that all the approaches for malware detection require modifications on the mobile device, which might void the manufacturer's guarantee.

SSA provides an infrastructure that allows the enterprise to apply malware detection techniques at the corporate network, independently of the marketplace from where mobile applications are obtained. This enables the enterprise to leverage the control on the mobile device by implementing the requirements they need in order to prevent insecure BYOD environments. To achieve this, the physical device is replicated and a virtual copy of it is obtained. This virtual copy is then sent to the corporate infrastructure. This replication includes the kernel, the device drivers, the user-space executable files, and all configuration files. Once the copy has arrived, the mobile device image is run in a software emulation and it is checked for security compliance, using the corporate security policies. During this testing, the authors state that both autonomous execution and triggered execution must be performed.

While the former type of execution does not require the input from the user, the latter does. Malware detection is then performed. The authors note that, depending on the malware detection technique, the virtual device might be changed in order to accommodate the needs of the technique. We must point out that this concept of recording and then replaying for mobile devices has been introduced by Portokalidis et al. [123].

Once all the testing has been finished, SSA provides different outcomes. First, it offers a passive outcome by notifying the user or the administrators about the results of the scan. Second, this framework allows active responses, which can either reset the phone to an uninfected state using an image stored at the corporate network, or by lowering the privileges of the device and preventing it from access sensitive data.

Titze et al. [122] recognize the limitations of this architecture. First, since the emulation cannot replicate the configuration of all mobile devices available in the market, the framework can only support a finite number of them (for example, some hardware characteristics such as finger print reader could not be supported). Second, the current implementation of SSA records the interaction of the user with applications, limiting the tests to mobile applications that were actually executed by the user. Additionally, the fact that user input is recorded leads to privacy concerns. Third, the malware detection technique used must be compatible with the emulation software, limiting the number of the techniques that can be used, or increasing the number of emulation that must be performed for each mobile device.

SSA is a solution that does not provide any form of space isolation, as the user's space is not separated from the corporate space. Corporate data protection is also not offered, since no encryption mechanism is found. Additionally, this framework might result intrusive to the user, as it records its behavior on the entire mobile device. However, SSA provides policy enforcement by analyzing the applications and

configuration on the mobile device. Further, it is not resource-intensive, as all the testing is performed in a emulated device located in the corporate infrastructure.

3.6.2 Commercial Solutions

In the previous section we have described framework solutions that have been proposed by the research community. Now we focus on frameworks that are commercial products These types of solutions are available in the market and are offered by well-known companies.

3.6.2.1 Cisco BYOD Smart Solution

Cisco's solution for BYOD environment [113] [124] is a set of network elements that provide the necessary infrastructure for providing a secure way to access the corporate network from the mobile devices. As a result, Cisco offers a modular and flexible architecture that adapts to the enterprise's needs.

First, Cisco provides solutions for wireless connectivity through the implementation of their connectivity products such as routers and wireless access points. These network infrastructures can further be managed with Cisco Prime Infrastructure Management software. Second, this BYOD solution provides policy management through the use of Cisco Identity Services Engine, which enables authorization, authentication. This engine also offers access policies based on the user's identification and the mobile device. Additionally, this system provides management tools to visualize who and what devices are accessing the wireless and VPN networks. Third, as mentioned before, Cisco offers AnyConnect, which provides a secure VPN access to the corporate network from other, untrusted networks. Finally, although Cisco does not provide any MDM solution, it can be integrated with third-party products.

By allowing the use of a third-party MDM, Cisco can leverage the enforcement of security policies. However, this also makes the solution intrusive to the user. Cisco BYOD Smart Solution does not offer any type of corporate data protection, as the use of cryptographic protocols is not provided to the mobile device. Further, space isolation is not achieved since this framework does not require any type of containerization on the BYOD device. Finally, this solution allows the implementation of this infrastructure without demanding any further resource from the mobile device.

3.6.2.2 Samsung KNOX

Samsung KNOX [125] [126] is a solution that includes enhanced security, container mechanisms, a cloud-based enterprise mobility management service, and a marketplace. This enhanced security is achieved with a trusted boot that guarantees that only verified and authorized applications can run on the mobile device. Additionally, Samsung KNOX relies on ARM TrustZone-based Integrity Measurement Architecture which checks the integrity of the Linux Kernel used in Android devices. Further, Samsung's solution provides Security Enhancement for Android which includes an additional third-party container.

Data isolation is provided by the implementation of containers. These containers are realized by developing secure zones on the mobile device for corporate application and by encrypting corporate data. First, IPSec and VPN are used for the cases where the data is transferred. Second, 256-bit AES is used when the data is stored, either in the internal memory or in SD cards. Additionally, communication between these two spaces can be controlled with the use of security policies. Samsung KNOX hypervisor is provided by INTEGRITY Multivisor technology [127].

The MDM solution that Samsung KNOX offers is part of a product called Samsung KNOX EEM, which is an alternative that works on the Cloud. This means that the

enterprise does not need to deploy any additional infrastructure in order to manage the user's devices. However, Samsung KNOX can be integrated with other MDM solutions such as AirWatch, Citrix, and SAP [128]. to enable the implementation of security policies.

Finally, since some applications can be insecure even if they are downloaded from a trusted marketplace, Samsung KNOX provides its own marketplace that guarantees the security of the mobile applications listed in it.

As any commercial product, Samsung KNOX has received certifications from external enterprises. The list of these certifications include The Common Criteria for Information Technology Security Evaluation, FIPS 140-2 Certification by the National Institute of Standards and Technology, and DISA MOS SRG Compliance from Defense Information Systems Agency (DISA), which is part of the Department of Defense of the United States of America.

The big disadvantage of this framework is that all the features are not available to all mobile devices. This means that, in order fully take advantage of the benefits that this solution offer, the user must own not only a Samsung device, but also a specific model from this company (generally the high-end devices from the Samsung galaxy line). While MDM features are compatible with iOS and non-Samsung Android, the marketplace, the container and the enhanced security are not available to iOS or Android devices from other brands.

Samsung KNOX achieves different goals for a secure BYOD environment. It offers space isolation by separating the personal space from the corporate space using containers. It provides corporate data security by implementing cryptographic primitives, even though it is not available to all mobile devices. Samsung KNOX provides policy enforcement by allowing integration with its own EEM or with a third-party MDM. As the containers are built-in on the Samsung devices, no extra resources

are needed and this solution is not resource-intensive. However, since it implements MDM capabilities on the mobile device, it results intrusive to the user. Further, personal data and corporate data are still stored on the mobile device, meaning that true isolation cannot be achieved.

3.7 Comparison of Solutions

We summarize all the solutions for securing BYOD environments in Tables 3.3 and 3.4. These tables includes all the solutions described in this chapter, their type, the goals they achieve, as well as their compatibility with Android and iOS OSes.

First, none of the solutions can satisfy all the goals. Moreover, solutions that share a type generally also address the same goals. Second, solutions that come from the research community only support Android, while commercial solutions provide support to both iOS and Android. Finally, most of the frameworks that come from the research community are not available to the enterprises.

		Desired Goals							
Solution	**Type**	Space Isolation	Security Policies	Corporate Data Protection	Non-intrusiveness	Low Resource Consumption	True Isolation	Android Support	iOS Support
INTEGRITY Multivisor [31]	MVM/FV	✓		✓				✓	
VMWare Horizon [33]	MVM/HD	✓		✓				✓	
L4Android [36], Cells [28]	MVM/L	✓		✓				✓	
AirWatch by VMware [40], Amtel MDM [41] [42] [43], BlackBerry BES10 [44], CA MDM [45], Citrix XenMobile Device Management [46] [47], Dell EMM [48], Good MDM [49] [50], IBM MaaS360 [51] [52], McAfee EMM [53], Microsoft Intune [54] [55], MobileIron EMM [56] [57] [58], SAP Afaria [59] [60], SOTI MobiControl [61], Symantec Mobile Management [62]	AB/MDM		✓	✓		✓		✓	✓

Table 3.3: Comparison between the different type of solutions based on their type, goals they achieve and support for mobile OSes. MVM represents Mobile Virtual Machine, FV represents Full Virtualization, HD represents Heavy Duty, L represents Lightweight, AB represents Agent-based, and MDM represents Mobile Device Management.

Solution	Type	Desired Goals: Space Isolation	Security Policies	Corporate Data Protection	Non-intrusiveness	Low Resource Consumption	True Isolation	Android Support	iOS Support
box [87], Dropbox [88], Google Drive [89], iCloud [90], OneDrive [91]	CB/C	✓			✓		✓	✓	✓
L2TP/IPSec [102] [103], PPTP [107]	VPN/S				✓		✓	✓	✓
Cisco IPSec [112], Cisco AnyConnect [113], Juniper Junos Pulse [114]	VPN/C				✓		✓	✓	✓
TEE [115]	TE	✓			✓	✓		✓	✓
TrustDroid [118]	TE					✓		✓	
MOSES [119]	TE	✓	✓	✓				✓	
2TAC [121]	F/R	✓	✓			✓		✓	
BYOD Security Framework [23]	F/R	✓	✓	✓					
SSA [122]	F/R		✓					✓	
Cisco BYOD Smart Solution [113] [124]	F/C		✓			✓	✓	✓	✓
Samsung KNOX [125] [126]	F/C	✓	✓	✓		✓		✓	✓

Table 3.4: Comparison between the different type of solutions based on their type, goals they achieve and support to mobile OSes (contd.). CB represents Cloud-based, C represents Commercial, VPN represents Virtual Private Network, S represents Standard, TE represents Trusted Environment, F represents Framework, R represents Research.

As mentioned earlier, none of the solutions presented in this chapter achieve the necessary solutions for a secure BYOD environment. In the following chapter, we introduce Remote Mobile Screen (RMS), a framework that achieves all the necessary goals for a secure BYOD environment.

Chapter 4

Remote Mobile Screen (RMS): Description and Experiments

Before we present our solution, Remote Mobile Screen (RMS), we start by describing BSF a framework solution, which is related to our work. Then, we provide a discussion on how RMS achieves all the goals for a secure BYOD environment. We provide the steps needed for a session initiation and termination. We analyze the features and challenges that RMS presents. Then, we describe an implementation of our framework that uses commonly available software. We perform a security analysis and identify security threats related to our solution. Further, we provide a security analysis of the architecture. Finally, we provide experimental results in order to address a set of these challenges.

4.1 BYOD Security Framework (BSF)

As mentioned in section 3.6.1.2, BSF is a framework presented by Wang et al. [23]. This framework has been designed to achieve three goals. First, space isolation is

required so that the personal space and the corporate space become separated from each other, and allow policies to be implemented for each of them individually. Second, corporate data protection is needed so that unauthorized access to this data becomes unfeasible, which is achieved by encrypting all the corporate data stored on the BYOD device. Lastly, security policy enforcement must be implemented so that the devices comply with the enterprise's requirements.

In order to meet these three requirements, BSF defines two entities: the enterprise side and the device side. The former is composed by all the corporate resources such as enterprise's servers, gateways to the Internet and the corporate data. In this side a Network Access Control (NAC) mechanism is in charge of providing access control when the BYOD devices try to access these resources. This access is either authorized or rejected based on the corporate policies. Additionally, the NAC has to differentiate between the requests from the personal space and the requests from the corporate space, which is achieved by implementing certificates for each of them. In order to manage the corporate policies a security policy database is deployed. These policies include information on how to handle the access request when it comes from a user space on a BYOD device, which devices are allowed to access the network, as well as the parameters of the connection. Finally, mobile devices are managed by integrating an MDM solution, which relies on the policy database and enforces these policies on the BYOD devices. Figure 4.1 shows a representation of all the components found in BSF.

At the device side, we can find space isolation between the personal space and the corporate space. Consequently, the personal space contains all the mobile applications and data owned by the employee, while the corporate space has the mobile applications and information needed by the enterprise. Since the corporate space must comply with the security policies of the enterprise, an MDM agent is installed

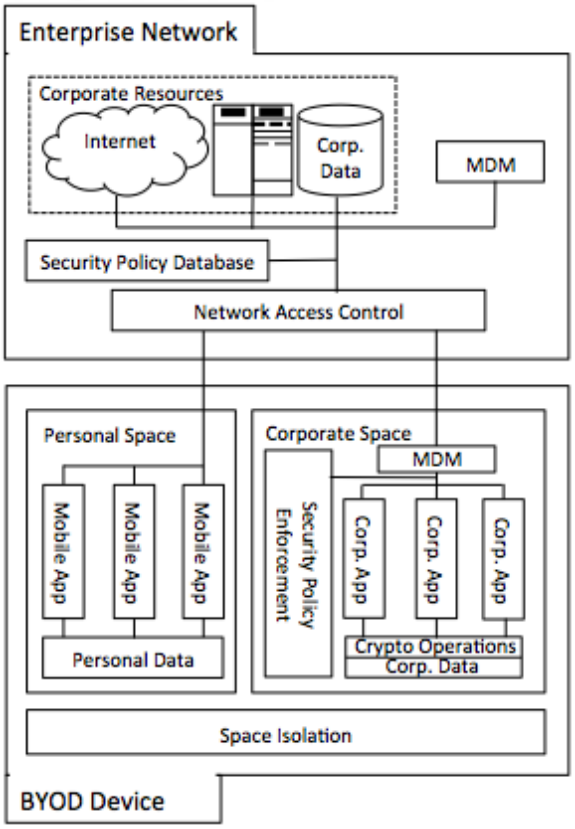

Figure 4.1: BSF architecture (based on [23]).

in this space, which provides the administrators management capabilities on the mobile device. Further, a security policy enforcement entity is also part of this space. These policies are stored in this space through the implementation of a security policy database. Finally, corporate data protection is achieved by implementing cryptographic algorithms as well as access mechanisms in order to prevent the data to be copied without proper authorization.

As defined by its own goals, BSF achieves corporate data protection by implementing cryptographic primitives and controlling the access to the information located at corporate space. It also supports security policy enforcement due to the use of MDM and security policy databases and enforcers. Finally, space isolation is achieved by separating the personal space from the corporate space. However, true space isolation

cannot be offered, as the corporate data is still stored on the mobile device. Further, in order to provide policy enforcement, BSF installs an agent that can become intrusive to the employee. Finally, space isolation is achieved using MVMs, but this could lead to a resource intensive solution.

4.2 Architecture presented by RMS

RMS modifies the BSF's architecture [23] by moving the corporate space located in the mobile device to the enterprise network. Additionally, RMS adds a new component that we denoted Corporate Space Manager, which is used to manage the access to mobile virtual machines located in the enterprise network. Finally, RMS uses the Virtual Network Computing (VNC) protocol (which is in turn based on the Remote Framebuffer (RFB) protocol [129]) to allow the user to access his or her proper corporate space. Just as in BSF [23], RMS presents a BYOD side and an enterprise side, which are described as follows.

4.2.1 BYOD side

Compared to the BSF [23], this part of the architecture is simpler. The mobile device only contains the employee's personal space. This means that the device cannot include any component with the exception of the personal data and applications of the employee. Consequently, there is no need to install either an MVM or an MDM agent on the mobile device. The only requirement of our framework is that VNC client application must be installed in the BYOD side. This client is used by the employee to access the enterprise space located at the corporate network. Part (A) of Figure 4.2 shows the BYOD side with all the mentioned components.

The novelty of our architecture relies on the way the employee access the corporate resources. In order to access these resources, the employee must install and use a VNC client from an app store such as Apple's App Store, Google's Play Store, or an app store provided by the enterprise. Then, the user does not access a desktop OS (i.e. Windows, Linux, Mac OS X), but he or she is presented with a mobile OS (i.e. Android, iOS). This requirement addresses the poor level of usability that desktop OSes have when they are accessed from a mobile device.

Desktop OSes are not designed to be scaled to the small screens that mobile devices present, since the graphical interface and desktop applications are developed for bigger screens. Further, desktop OSes do not implement gestures (e.g. pinch-to-zoom, swipe, etc.) nor the type of input (i.e. finger or stylus) customary to a mobile OSes. Consequently, when the employee accesses the enterprise side, he or she is presented with an interface designed for a mobile device. To our understanding, the concept of a mobile device that access a mobile OS over a network has not been used in BYOD environments, or for any other kind of purpose.

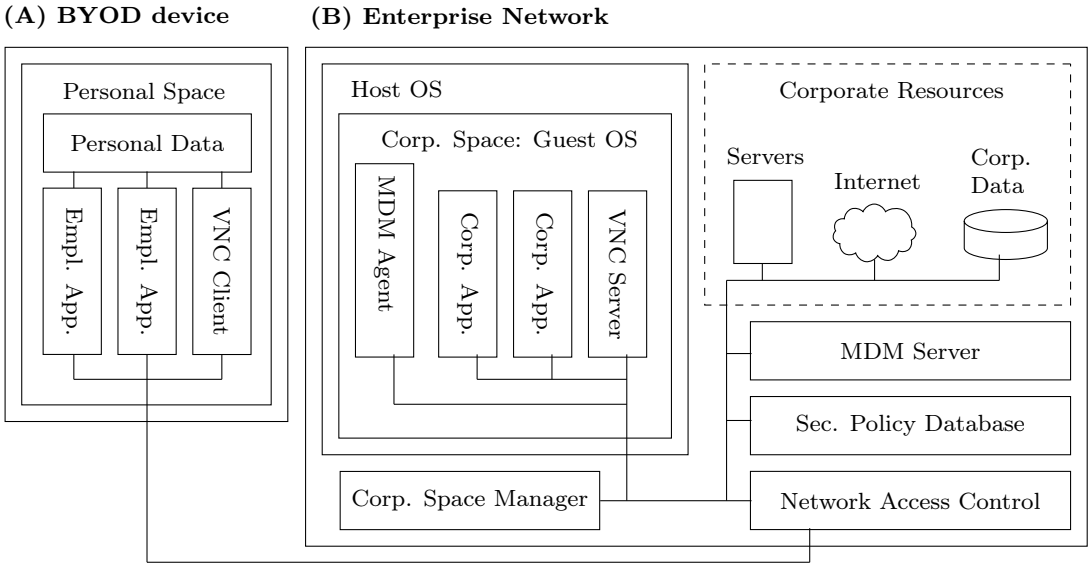

Figure 4.2: RMS architecture.

4.2.2 Enterprise Side

This side is composed of most of the elements needed in the RMS architecture. This is because the enterprise side does not suffer from the limitations in terms of resources that a mobile device has. As a consequence, in the enterprise side we can find the Corporate Resources, a Network Access Control, a Security Policy Database, an MDM server, Corporate Spaces and the Corporate Space Manager. The enterprise side, as well as its components, is depicted in Part (B) of Figure 4.2

The corporate resources are composed by devices and services such as e-mail servers, web servers, gateways to the Internet, or any proprietary application that the enterprise has. The Network Access Control is in charge of authenticating valid employees and authorizing them to access the enterprise resources. The Network Access Control not only analyzes requests from outside of the enterprise network, but also evaluates the requests that come from inside of the enterprise network. The Security Policy Database stores all the policy definitions that the enterprise has. In order to grant or reject access, the Network Access Control relies on the security policies that the Security Policy Database contains. The MDM server is in charge of enforcing all the security policies on the Corporate Spaces.

At the enterprise side we find the corporate space, which contains all the corporate data and applications that the employees needs for working. We can define this space as a VM provided with a mobile OS. Consequently, the enterprise side contains a server running a VM software that deploys several corporate spaces in the form of guest OSes. Each of these corporate spaces is assigned to one employee only. Further, each of the mobile OSes has installed a VNC server, which is configured in such a way that the employee can access his or her corporate space using the VNC client found in his or her BYOD device. In addition, each corporate space is provided with

an MDM agent, which is in charge of implementing the security policies enforced by the MDM server.

The Corporate Space Manager is in charge of administrating all the corporate spaces in the enterprise network. This manager is able to create, delete, start, pause, resume and stop the VMs in the VM server, providing elasticity to the architecture. The Corporate Space Manager operates as a proxy server, since it inspects the content of VNC packets and performs actions based on such content. In order to decide whether to create, start or resume an existing VM, the Corporate Space Manager must keep track of which VM is assigned to each user. Further, it is also in charge of setting the networking configuration of the VM, as well as adding and removing Network Address Translation (NAT) entries in the VM software so that the employees' requests get forwarded to the corresponding VM.

Further, the Corporate Space Manager can be implemented as a centralized entity or deployed as a set of geographically distributed instances, based on the needs of the enterprise. This second configuration is particular beneficial in large-scale enterprises. The advantages of a distributed network is that a Corporate Space Manager can be deployed closer to the employees thus reducing the latency in the communication. We discuss issues related to latency in Sections 4.7.8 and 4.8.3. Deploying a distributed network implies solving problems similar to the ones found in Content Distributed Networks such as the Object Replication Problem, and the Request Routing Problem [130]. However, we must point out that in the case of RMS, the employee is not just accessing content, but also modifying such data.

4.3 Session Initiation and Termination

RMS offers the employee access the to corporate resources just as if he or she were using an MVM installed in the BYOD device, but without affecting its resources. Following, we present how an employee accesses his or her corporate space.

When an employee wants to access his or her corporate space, the VNC client is started in the BYOD device, and a connection to the enterprise network is requested to the Network Access Control. The Network Access Control then authenticates the employee using a combination of username and password and forwards the VNC connection request to the Corporate Space Manager. The manager checks the status of the VM assigned to the employee. If the VM does not exist, the manager creates it; if the VM is stopped, the manager starts it; If the VM is paused, the manager resumes it. Then, the Corporate Space Manager sets up the NAT entries and forwards the VNC connection to the VM. Once the VNC connection request reaches the VNC server, the session can be established.

After the session has been established, the VNC client application is in charge of sending the gestures input at the BYOD device to the VNC server in the corporate space. Further, the VNC client is in charge of obtaining screen images from the VM and presenting them to the employee using the screen on the BYOD device. For example, if the employee taps on an application in the corporate space, the VNC client sends the tap to the VNC server and requests a new screen image. Then, the user is presented with the image of the application he or she wanted to use.

Finally, once the employee wants to finish the connection to the corporate space, he or she closes the VNC client in the BYOD device. The VNC protocol is a stateless protocol, and no VNC message is sent to the corporate network. However, since VNC run on top of TCP, the Corporate Space Manager can detect the session teardown,

then remove the NAT entry and finally pause the proper VM. In the case of a re-connection the VM can be resumed from a paused status, and the employee can continue using the VM from the point where he or she left it previously.

4.4 Features Offered by RMS

RMS achieves all the desired goals for a secure BYOD environment. It provides separation of spaces by implementing the corporate space in the enterprise network while leaving the personal space at the mobile device. This way, the enterprise can only focus on enforcing security policies in the corporate space.

The way the corporate space and the personal space are separated achieves the goal of true isolation, since both spaces are never shared either in the BYOD device or in the VM. Even if the BYOD device gets lost of stolen, the corporate space is not compromised because no enterprise data is stored in the mobile device.

This isolation of the data also guarantees data protection, as the corporate data remains always in the enterprise network. Further, VMs offer containers that separate the corporate spaces of different users. This prevents data leakage between corporate spaces. Additionally, the implementation of the Network Access Control and the Corporate Space Manager provides a strict control of the information's flow between personal and corporate spaces.

RMS implements an MDM agent in the corporate space of each employee. This, along with a MDM server, allows the enterprise to enforce security policies. For example, instead of installing applications from the Internet in the guest OS, the employee is offered a curated, authorized list of applications allowed by the company. Furthermore, the enterprise can decide which permissions an employee can have in the corporate space. RMS offers and additional advantage because all the security

efforts are concentrated on a single point in the corporate network, instead of focusing the security efforts on every mobile device.

The proposed framework is not intrusive to the employee because the BYOD side does not require any application with administrative or root privileges as an MDM agent might request. Further the personal space cannot be controlled or monitored by the enterprise since the latter does not establish any type of communication with the mobile device. In addition, the enterprise does not need to provide support to the BYOD at all, since there is no software installed by the corporation.

In order to provide space isolation, a VM has been implemented at the corporate network. This type of solutions are resource demanding and the performance of mobile devices can be negatively affected. RMS benefits from the fact that the VM is not deployed on the BYOD device in order to achieve the goal of low resource consumption. Further, the employee does not need to allocate storage space on his or her device for the corporate data and work-related applications.

Another benefit of this framework is the fact that it is platform-agnostic, as it can provide service to every mobile platform that has a VNC client application. For example, RealVNC is available on Android, Blackberry, iPhone, Symbian and Windows Mobile platforms [131]. Further, since only a VNC client is needed, BYOD devices can be changed by the employees without consulting the enterprise. This provides flexibility and deals with the fast high frequency of obsolescence in an efficient way.

Finally, the enterprise can avoid OS fragmentation by implementing RMS, as the enterprise can decide to deploy a specific version of a mobile OS in each of the VMs. Consequently, it becomes easier to provide support to the VMs, as they are similar. Moreover, enterprise does not need to consider the situations where one productivity software is available for some versions of an OS, but unavailable to others.

4.5 Implementation

We developed a proof-of-concept implementation with commonly available software. First, in order to deploy the mobile OS we used different type of servers running either Linux or Mac OS X. The VM software installed in these servers was Oracle's VirtualBox [132], which is an open-source virtualization software.

In order to realize the corporate space, we deployed AndroVM [133] and Android-86 [134] as guest OSes. AndroVM is an open-source implementation of Android for the x86 architecture that is distributed in VirtualBox-compatible files, known as .ova files. This feature makes AndroVM easier to deploy. However, AndroVM was developed in such a way that the boot configuration of the operating system is located in a partition that cannot be directly modified by the user, which makes harder any form of special configuration. Android-x86 can be modified with less restrictions. For example, the screen resolution of the mobile OS can be easily re-sized. Android-x86, nevertheless, is harder to deploy as it is distributed as a live CD that must be then installed on the VM. It is important to note that there are no iOS implementations that run on VirtualBox for two reasons: first, it violates the iOS's Software License Agreement [27] and second, there is no port of iOS to the x86 architecture.

The VNC server used is Droid VNC server [135] but modified for the x86 architecture, as the one available for the ARM architecture presents issues for binding the ports at the guest OS. Droid VNC Server is open-source and presents features such as password authentication, rotate/scale, and clipboard support. One of the requirements for Droid VNC Server to work is root privilege on the device.

Finally, for VNC clients, we have employed the free versions of Mocha VNC Lite [136], VNC Viewer [137] and VNC Client - Universal App [138] for iOS, Mocha VNC Lite for Android [139], and TinyVNC [140] for Windows Phone. In all these VNC

clients we have found that, since they are not designed for accessing a mobile OS on the server side, they lack features such as pinch-to-zoom and swipe. Moreover, every time pinch-to-zoom is issued, the VNC clients zooms the image gotten from the VNC server. Further, swipe gestures lead to move the image displayed at the client.

Figures 4.3, 4.4 and 4.5 show our implementation. In these figures we can see three mobile devices running at the left side of each picture and three virtual machines running the corporate space at the right side of each picture. In Figure 4.3, Android-x86 is displayed on the screen of an Apple iPhone 5s running iOS 8 and Mocha VNC Lite as a client. In Figure 4.4, an Apple iPad Mini running iOS 8 and Mocha VNC Lite access an instance of Android-x86. Finally, in Figure 4.5, AndroVM is used as the corporate space for a Samsung Focus with Windows Phone 7 running TinyVNC as a client.

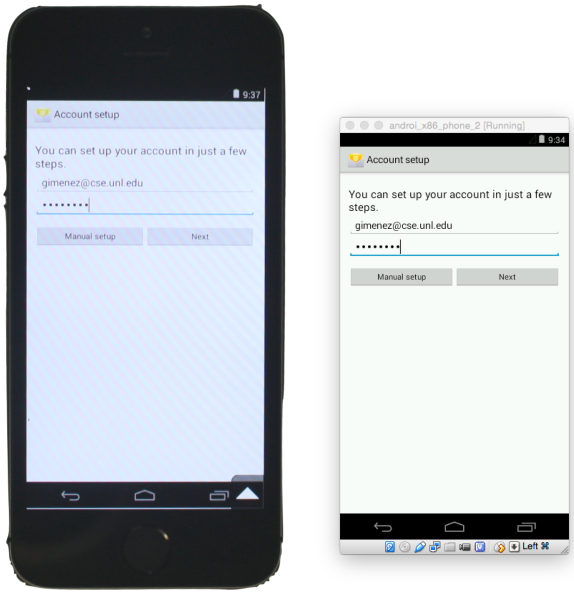

Figure 4.3: Implementation using an Apple iPhone 5S accessing Android-x86 as corporate space.

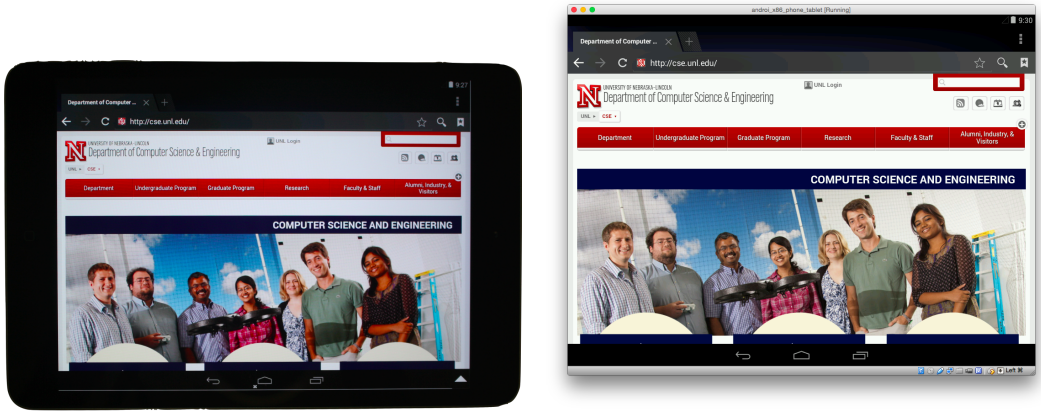

Figure 4.4: Implementation using an Apple iPad Mini accessing Android-x86 as corporate space.

Figure 4.5: Implementation using a Samung Focus accessing AndroVM as corporate space.

4.6 Security Analysis

We have identified a set of threats that might affect the security of our framework. Firstly, the RFB (and consequently VNC) protocol lacks an acceptable level of security. Therefore, the goal of Corporate Data Protection cannot be guaranteed. Richardson et. al. [129] state that this protocols only offers a cryptographically weak

password authentication mechanism, and it does not provide any form of protection against passive or active attacks. In order to overcome this issue, the researches mention that any RFB connection must be set up on top of a VPN connection, such as the one that SSH and IPSec provide.

Secondly, the RFB protocol allows sharing the clipboard and files between the client and the server. Generally, this characteristic would be considered a feature, but in our framework this might prevent space isolation between the personal space and the corporate space. Consequently they threat the goals of Space Isolation and True Space Isolation. Disabling both features from the server side can prevent this threat. This way, both clipboards are confined to their respective spaces and file-sharing is not allowed. Another approach is to enable the flow of information between spaces only in one direction. For example, an employee can only upload files to the server, or send information from the client's clipboard to the server's clipboard. In this situation the employee cannot download files from the corporate space, nor obtain the information from the clipboard at the corporate space.

Thirdly, there might be situations where the content of the screen of the BYOD device is captured, for example by taking screenshots of the device's screen or when an attacker looks over the shoulder of an employee to see the content of the mobile device (i. e. shoulder surfing). While taking screenshots might affect the goals of Corporate Data Protection and Space Isolation, shoulder surfing only affects the goal of Corporate Data Protection. There is no technological solution to prevent these situations, other than privacy and screen protectors against shoulder surfing. Since the company does not have any software installed in the mobile device that could prevent screenshots, the only alternative against this threat is employee education, where the employee learns about the different threats that put the corporate information at risk, and the precautions that must be taken in order to mitigate these

threats.

Lastly, Denial of Service is an important threat that targets the corporate network. In this type of attack, the corporate resources are requested to perform a high volume of tasks (either processing or networking) such that the performance of the services get degraded or completely interrupted. Consequently, employees cannot access their corporate spaces and cannot work properly. These problems are addressed by using redundancy of single points of failures and backing up data. For RMS, the Network Access Control and the Corporate Space Manager must present a fail-over system, and the VMs must be frequently backed up.

4.7 Challenges Presented by RMS

Our framework presents a series of challenges that must be addressed. We identify these challenges and discuss how they affect RMS.

4.7.1 Connectivity

In order to access the corporate network, each employee must have access to a network such that he or she is able to establish a connection to the enterprise network. This requirement is part of the trade-off that RMS creates in order to achieve all the goals needed for a secure BYOD environment. Consequently, the mobile device heavily relies on continuous connectivity. Without a connection the employee is not able to perform work-related actions such as sending and receiving work-related emails.

However, we consider that given the high level of connectivity found in mobile devices (either cellular networks or WiFi), the employees should not have issues when it comes to accessing the corporate network.

4.7.2 Android and the x86 architecture

Android is an operating system developed and optimized for the ARM architecture [141]. Since the servers we used do not present an ARM architecture but an x86 architecture, the virtualization software also offers an x86 architecture to the guest OSes. Consequently, the version of Android used in our implementation must be compatible with the x86 architecture.

In our research, we have found and tested three implementations of Android for the x86 architecture: AndroVM [133], Genymotion [142] and Android-x86 [134]. The first one is an open-source implementation that was later merged into Genymotion and it is no longer available. Genymotion is also an open-source implementation that presents additional features to the ones that AndroVM presents. The goal of both AndroVM and Genymotion is to provide a efficient testing framework for application development. Finally, Android-x86 is an open-source port of Android to the x86 architecture. The goal of the community that develops Android-x86 is simply to support the mobile OS on different x86 platforms. On the one hand, in our testing both AndroVM and Android-x86 proven to work with VirtualBox. On the other hand, Genymotion implementations presented issues once the mobile OS were run from the VirtualBox console instead of using the manager provided by Genymotion.

4.7.3 Usability

RMS presents a series of drawbacks when it comes to usability mainly based on the adoption of RFB. This protocol considers two main events: input events (iė. clicks) and output events (iė. screen images). Even though RFB has been updated over the time to support new input events such as scroll-wheel events, the protocol has not been designed nor updated for the mobile devices, and it lacks support for gestures

that are customary in such devices. Further, as mentioned before, VNC clients do not use these gestures at the mobile device in a proper way. For example, if a user tries to scroll the content of a web page, the client will interpret that the user is trying to access parts of the screen that are not shown and move the image received from the server out of the screen bounds.

In order to overcome this issue, we propose three alternatives. The first one is to extend the RFB protocol in order to include these new types of input events. Under this situation, the VNC Server and the VNC clients must be also updated to support these features and they must implement gestures such as swipe and pinch-to-zoom properly.

The second alternative is to migrate from the RFB protocol to another protocol such as Remote Desktop Protocol (RDP) [143], a proprietary protocol developed by Microsoft, which provides extensions to the ITU T.120 family of protocol. According to Microsoft, this protocol implement touch gestures [144]. In our research we did not find any RDP server for Android. VirtualBox, however, supports a VirtualBox Remote Desktop Protocol that can be installed as part of the extension pack. The implementation of this protocol involves the adoption of RDP clients by the employees. Finally, the fact that this protocol is proprietary may discourage some enterprises to use it.

The final option is to consider that the RFB protocol presents enough drawbacks as an argument for completely avoiding it use. Consequently, a new and modern communication protocol could be developed considering all the features that are missing in RFB as well as other features that could be leveraged for RMS. In the list of features we can include touch gestures, security for both passive and active attacks, implementation of notifications from the applications in the corporate space to the BYOD device. In this scenario, not only the communication protocol would have to

be developed, but also a server and clients for each of the mobile platforms.

4.7.4 Video Playback

It results impractical to watch videos over VNC for two reasons: first, there is no audio available; and second, the refresh rate of VNC is not fast enough to present continuity in the video playback. We have tried different encoding standards for VNC and we can see an increment in the refresh rate. However, the refresh rate of the images is not fast enough for a seamless video playback.

However, we can find research on improving the frame rate of the VNC protocol. For example, Tan-Atichat et al. [145] shows that up to 14 frames per second can be achieved in environments with 500 milliseconds of latency. Even though this is a good improvement, smooth video playback requires between 24 and 30 frames per second.

4.7.5 Malware

Threats related to malware must be considered for both the personal space and the corporate space. First, since the enterprise does not have any control over the BYOD device, it cannot enforce the use of an antivirus application for the personal space. The enterprise, however, can educate the employee about the risks related to malware and explain the consequences of malicious software. Moreover, since the framework presents a Network Access Control, malware attacks to the corporate network can be mitigated.

Second, at the corporate space the enterprise has full control over it. Consequently, it can install any antivirus software for mobile devices, and it can enforce policies that prevent the employee from remove or stop the antivirus.

4.7.6 Compatibility

Android provides a Dalvik register-based VM, which is based on Java [146]. This way there is an abstraction layer between the Android kernel and the mobile applications. Even though Android is mainly developed for the ARM architecture, the use of a VM provides portability, as mobile applications can be ported to other implementation of Android running on different architecture.

For RMS, the use of the Dalvik VM should provide sufficient compatibility such that applications from Google Play can be seamlessly installed in an x86-version of Android. Nevertheless, Google also offers an NDK [147] which takes advantage of native libraries offered for ARM, x86 and MIPS architecture, offer better performance and allow the development of applications in C and C++. Since most of the mobile devices are implement an ARM processor, it becomes clear that applications that need better performance will use native libraries for the ARM architecture. As a consequence, the compatibility of application can be broken when they are installed on the x86 architecture.

According to Choi et al. [148] around 75% of the applications found in Google Play do not present any native ARM code. However, the remaining 25% must be addressed. In order to overcome this situation, Intel released a binary translation library called Houdini [148], which allows applications compiled for the ARM architecture to run on x86 architectures. On the one hand, the distributions of AndroVM and Android-x86 include this library. Genymotion, on the other hand, does not include this library and it must be installed separately.

In section 4.8.1, we address this concern by carrying out a survey on the productivity applications available in Google Play for each of AndroVM, Android-x86 and Genymotion.

4.7.7 Scalability

The framework we propose only allows one employee per VM, as each employee needs one mobile OS running in the enterprise network. This means that each VM requests resources from the VM server that cannot be used by other purpose. Consequently, there might be concerns regarding the scalability of the RMS framework. In this section we discuss them and propose solutions to decrease the resources needed by this framework. Later, in section 4.8.2, we provide the results of scalability tests that were performed on RMS.

Once a new VM is created a list of available resources (e.g. RAM memory, hard disk capacity, video memory) must be set. For example, AndroVM requires 1024MB of RAM, three hard drives of 541MB, 5.51GB and 8.00GB and 8MB of video memory. However, these resources are the limit these virtual devices can use, meaning that when they run, they do not use all the allocated resources and leave them available so that the server can allocate them to other VMs or other processes. As a consequence, a server is able to run more VMs than initially expected, and a single server with standard configuration can run several VM, providing service to several employees

Moreover, the VM software has the feature of saving a machine state instead of turning it off, while releasing resources at the same time. This feature has two benefits: it saves unused resources for other employees, and it provides a fast start after the employee has disconnected and re-connects. Consequently, any scalability test must not be performed on the total available employees that use RMS, but on the concurrent number of employees.

4.7.8 Latency

RMS is a real-time solution, as the employees expect a prompt response from the corporate space. Just like every real-time solution, RMS is sensitive to the delay in the connection between the VNC client and the VNC server. We define latency as the time between the user's input and the response obtained from the VNC client application. This considers the processing delay at both ends, the propagation delay over the links between the employee and the network, as well as the queuing delay proper of communication networks.

According to Shneiderman [149], the response time for typing, cursor motion, and mouse selection must be in the interval between 50 to 150 milliseconds. Consequently, a real-time application that requires interactions from the user must meet this requirement so that the user does not become frustrated with the application.

In general, the farther away the user is from the corporate resources, the bigger the latency will be product of the propagation delay and the queuing delay. Additionally, the more concurrent employees using the application, the more latency they will experience product of the processing delay.

One solution that addresses this issue is to migrate RMS to the Cloud, such that a distributed set of servers running the corporate spaces could be allocated close to the employees. Furthermore, by implementing a group of servers, the number of concurrent employees accessing a server will decrease, which will lead to a decrease in the processing delay.

Additionally, the use of the Cloud could lead to other benefits such as redundancy and higher availability of the resources that guarantees a continuous service to the employee. Even if the entire datacenter becomes unavailable, another datacenter can be used at the expense of a higher latency. Lastly we can mention the fact that Cloud

solutions offer elastic computing by allocating resources on demand. For example, if one server is running low on available resources, a back-up server can be enabled to balance the load. In our experience, we have found that implementing VirtualBox on top of a virtual machine in the Cloud is not straightforward. However, we were able to run the virtualization software on one of Amazons EC2's windows servers, albeit with limited performance.

In section 4.8.3, we address the latency issue by providing results of experiments for both a centralized solution and a distributed alternative.

4.8 Compatibility, Scalability and Latency Experiments

In this section, we describe and show the results of experiments related to the challenges that RMS presents. The experiments are a compatibility test of productivity applications found in Google Play for different implementations of Android for the x86 architecture, a scalability test to determine how a server performance varies as the number of concurrent users increases, and a series of latency evaluations in order to describe the behavior of our framework in a large-scale deployment.

4.8.1 Compatibility

Google Play classifies the available applications different categories, being one of them the productivity category. In order to determine the compatibility of RMS with the applications, we tested a list of productivity applications suggested by Google Play. This list included three paid applications, and did not include any mobile application from Microsoft. As a consequence, we did not test the paid applications, but we did

test four applications developed by Microsoft. Each of these mobile applications was analyzed on a VM running AndroVM on a smartphone with Android 4.1, a second VM running Android-x86 on a tablet with Android 4.4, and a third VM running Genymotion on a tablet with Android 4.4. Tables A.1 and A.2 from Appendix A show the list of the mobile applications that were tested, as well as a description of each of them.

Our results indicate that for AndroVM 37 (90.2%) mobile applications can be installed, and 33 (80.4%) work without issues. In the case of Android-x86, 40 (97.6%) mobile applications can be installed, and 33 (80.4%) work without issues. Finally, for Genymotion 37 (90.2%) mobile applications can be installed, and 35 (85.4%) work without issues. Figure 4.6 shows a graphical representation of the results for this experiment.

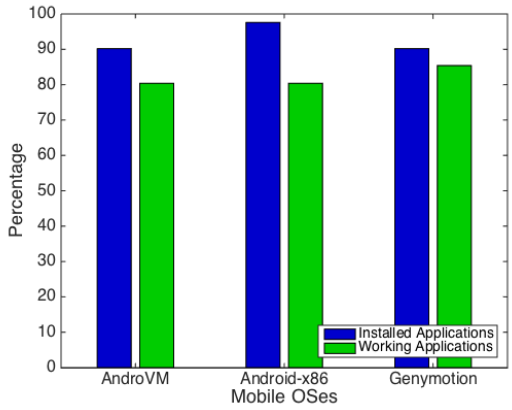

Figure 4.6: Results of compatibility experiment.

The applications that presented issues are “Papyrus - Natural notetaking” and “TeamViewer for Remote Control”, which were installed in all the VMs, but they did not work for AndroVM nor Android-x86; “Quip: Docs, Chat, Spreadsheets” was installed in all VMs but it not work properly for any of the VMs; “Echo Notification Lockscreen” and “Slack” were not found in Google Play for AndroVM and they were

not installed nor run; the presentation feature of "WPS Office + PDF" did not work for Android-x86; "Microsoft Office for Mobile" was not found in Google Play for each of the VMs, and it was not installed nor run; "Microsoft OneNote" was not found in Google Play for AndroVM and Genymotion, and it was installed in Android-x86 but it did not run properly.

4.8.2 Scalability

One of the challenges of RMS is scalability, as one VM is needed for each BYOD device. Our scalability tests consist on running our implementation of RMS in such a way that we increase the number of concurrent VMs over the time, until the system becomes unresponsive.

In order to provide context an MDM solution that supports up to 5000 users requires three virtual machines [47]: an XM Device Manager with 2 to 4 virtual CPUs, 4GB of RAM and 24GB of disk space, a Secure Mobile Gateway with 2 virtual CPUs, 2GB of RAM and 24GB of disk space, and a XM SQL Server with 2 virtual CPUs, 6GB of RAM and 24GB of disk space.

For our experiment, we worked with a high-performance server from Holland Computing Center. The characteristics of the server include a AMD Opteron(TM) Processor 6272 with 64-cpu divided in 4 sockets of 8 cores each, with an x86_64 architecture running at 2.16GHz; 256GB of DDR3 RAM in 32 modules of 8192MB each, running at 1600 MHz; a Hitachi HUA72201 hard drive of 1TB, with a cached read speed of 3583.09 MB/sec and a buffered disk read speed 127.20 MB/sec. The OS of this server was Ubuntu 14.04.1 LTS, with a 3.13.0-44-generic kernel. Additionally, the server did not use any swap memory.

The implementation of RMS includes VirtualBox 4.3.20r96996, running AndroVM

with 1024MB of RAM for each VM.

The statistics we used to evaluate our experiment are presented as follows:

- Memory consumption: how much memory the entire server is using. This parameter is obtained from command `free`.
- CPU usage: the percentage of CPU utilization of the entire server. This parameter is obtained from command `top`.
- Number of Threads: the amount of threads the server is running. This parameter is obtained from command `ps`.
- Load: average the load the system is experienced for the last 15 minutes. This parameter is obtained from command `top`.
- Start Time: time between a command for starting a mobile OS is issued and until the mobile OS answers a ping echo request. This parameter is obtained as a post-process event.
- Restart Time: time between a re-start command is issued (for a paused VM) and when the processing of such command ended. This parameter is obtained as a post-process event.

Since the server is configured in such a way that it will always allocate new memory for new VM by killing other processes, the server generally kills a VM in order to continue working. This means that we cannot obtain the maximum amount of concurrent VMs based on the moment the server crashes. Consequently, we obtain the limit of maximum concurrent mobile OS when the value of Start Time is greater than 45 seconds, or when the value of Restart Time is greater than 25 seconds. An-

other alternative is to set the limit when the number of threads decreases by a specific number, which would indicate that the system killed a VM.

A script in BASH was developed in order to perform this experiment. The statistics were measured by the script before the first VM was started and then after each VM is started. The scripts used in this experiment are shown in Appendix B. We have run this experiment four times. For each of the experiments, the system became unresponsive at 599, 597, 594 and 596 VMs.

The behavior of memory usage is presented in Figure 4.7. Our results show that when no mobile OS is running we have on average a memory consumption of 2.1GB. Then, for each mobile OS, we add 0.67GB on average until we reach the total available memory of 256GB. This limit is approximately around 380 mobile OS. After this point, the memory consumption remains close to 256GB.

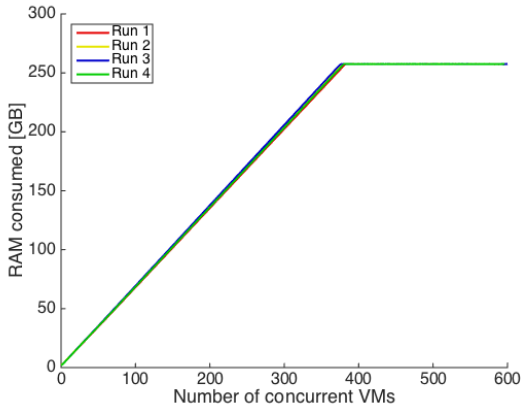

Figure 4.7: RAM usage as a function of the number of concurrent VMs.

The CPU usage presents a behavior shown in Figure 4.8. As expected, we can see how the usage of the processor increases as the number of concurrent mobile OSes increases, with spikes over 25%. In order to better describe the behavior of this parameter, we used the fitting toolbox provided by Matlab to obtain a simple, linear equation for the first run. The result is shown as follows:

$$CPU = 20.35 \times 10^{-3} N + 3.13 \tag{4.1}$$

where CPU is the CPU usage and N is the number of concurrent VMs. Consequently, we can observe that every new VM adds on average 0.02% to the total CPU usage.

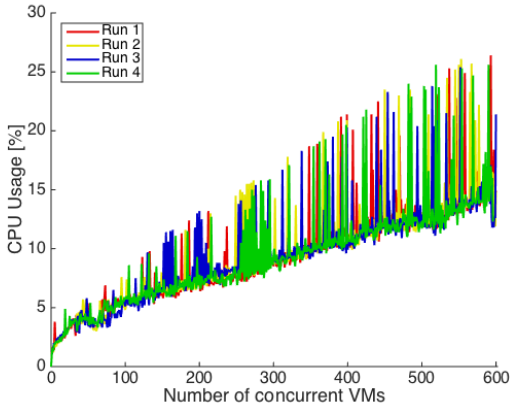

Figure 4.8: CPU usage as a function of the number of concurrent VMs.

For Load, we show its behavior in figure 4.9. As expected, we see that this parameter increases as the number of concurrent VMs increase. The behavior of this curve is polynomial and it can be approximated with the following equation:

$$Load = -5.03 \times 10^{-6} N^3 + 5.33 \times 10^{-3} N^2 + 251.39 \times 10^{-3} N - 9.69 \tag{4.2}$$

where $Load$ is the average load of the system for the last 15 minutes and N is the number of concurrent VMs.

We present the results of the number of threads as a function of the concurrent number of VMs in Figure 4.10. The number of threads increases linearly, as it can be shown in the following regression formula:

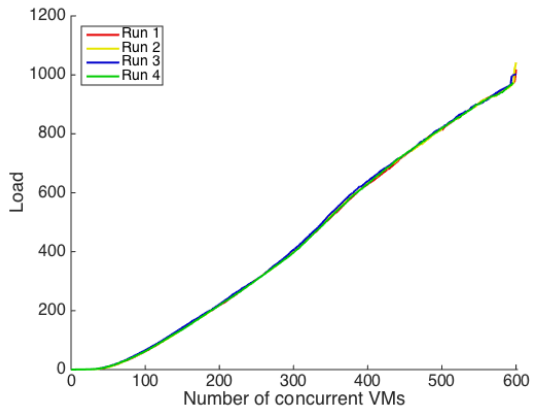

Figure 4.9: Load of the system as a function of the number of concurrent VMs.

$$Threads = 20.05N + 764.05 \tag{4.3}$$

where $Threads$ is the number of threads in the system and N is the number of concurrent VMs. Consequently, any new VM requires 20 threads.

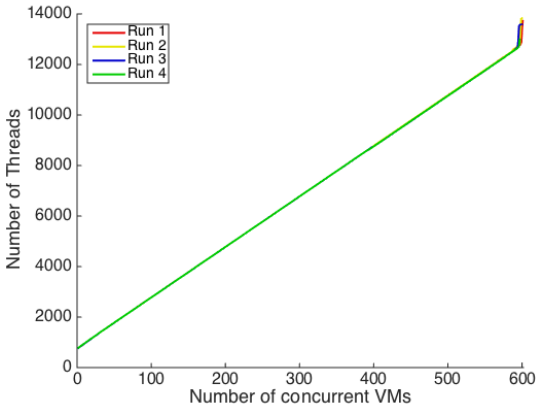

Figure 4.10: Number of threads as a function of the number of concurrent VMs.

We present the results of our experiments for Start Time in Figure 4.11, which shows how this time varies as the number of concurrent VMs increases. Further, we can notice that there is a strong increment in the start time when the system reaches the limit of concurrent VMs. We can represent the behavior of the Start Time using

the following polynomial regression formula:

$$ST = -357.68 \times 10^{-9} N^3 - 243.91 \times 10^{-6} N^2 + 65.03 \times 10^{-3} N + 5.79 \qquad (4.4)$$

where ST is the Start Time from power-off status and N is the number of concurrent VMs.

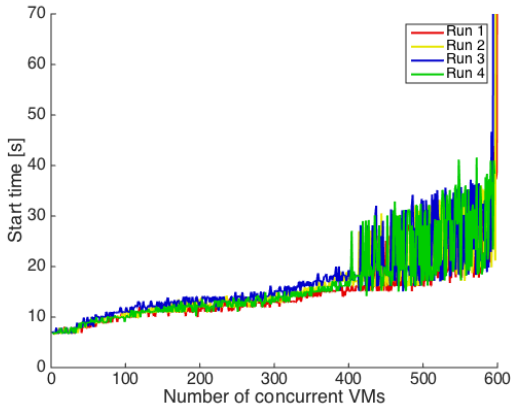

Figure 4.11: Start Time as a function of the number of concurrent VMs.

Finally, for Restart time, we show its behavior in Figure 4.12. The value of Restart Time remains between 0 and 2 in most cases, but it presents a spike once the server reaches the limit of supported VMs.

From these experiments we can see that the number of concurrent VMs depends on the amount of available RAM. Consequently, we limited the amount of RAM in the high-performance server to 128GB, 64GB, 32GB, 16GB, and 8GB. In each case, the server presented a behavior consistent to the limit in the RAM. We repeated four run for each of these cases and performed a regression in order to estimate the number of concurrent VMs as a function of the available RAM. Such function is presented as follows:

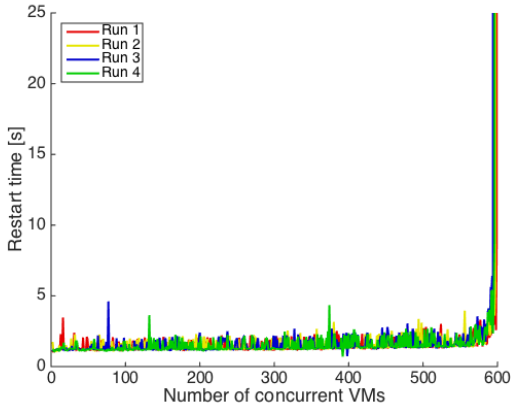

Figure 4.12: Restart Time as a function of the number of concurrent VMs.

$$N = 2.31RAM + 4.70 \tag{4.5}$$

where N is the number of concurrent VMs and RAM is the available RAM in the system in GB. From this formula, we can see that each GB of RAM allows using 2.3 VMs. For completeness, we show in Figure 4.13 the representation of the data points and the regression formula.

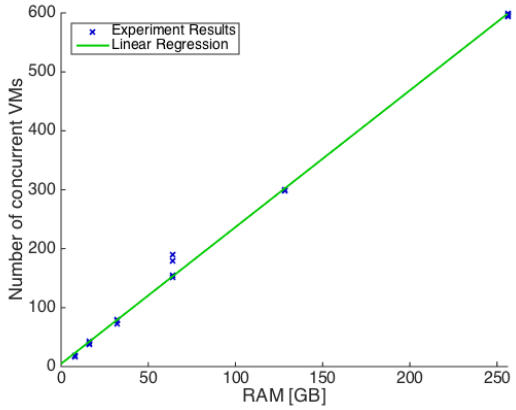

Figure 4.13: Number of concurrent VMs as a function of the available RAM.

From these series of experiments we can draw two conclusions. First, the amount of RAM that a VM has allocated is not factor in determining of how many concurrent

VMs we can support. The factor that determines limit for the number of VMs is the amount of RAM that each VM consumes. Second, given the available RAM in the server running the virtualization software, it results easy to estimate the maximum number of concurrent VMs that the server can support, since this relationship is linear.

4.8.3 Latency

RMS is delay-sensitive and a high level of latency impacts its usability. Consequently, we analyzed how the system behaves in a scenario where the users are distributed across the country and try to access their corporate spaces. We have designed a series of experiments in order to measure the effect that the number of concurrent users and the ping delay have over the overall application delay.

For these experiments we set up servers and clients in GENI [150], a distributed testbed with several nodes across the country. The servers were configured with our implementation of RMS using VirtualBox and AndroVM. The clients were using vncdotool [151], a command line VNC client that can execute automated commands. Each client was configured to send 12 clicks to the server and receive 12 screenshots from the server.

In order to obtain the application delay, we capture VNC packets at each of the clients. Then, we define the application delay as the time difference between a packet with a click message has been sent from the client and a screenshot is completely received by the client (including re-transmissions). Consequently, the application delay presented in this section does not include the time needed for processing. In order to set up these servers, we used the scripts shown in Appendix C.

4.8.3.1 First scenario

The goal of our first experiment is to determine how the number of concurrent clients affects the application delay. For our set up, we selected one client from the University of Indiana and a server at Georgia Tech. We first measure the application delay when there are no other clients, and then we added a second client for a different location and run the experiment again to obtain the application delay in the Indiana node. We continued this process by adding clients and measuring the application delay up to 15 clients.

We define N as the number of concurrent clients that connect to a server and k as the number of servers. Consequently, for this scenario we have $N = \{1, 2, 3, \ldots, 15\}$ and $k = 1$.

The list of clients we used and their locations are shown in Table 4.1. A graphical representation of the location of the clients (in light blue), as well as the server (in red) is presented in Figure 4.14. This figure also shows the order in which the clients were added to the experiment.

Figure 4.14: Distribution and order of the clients across the country.

After the experiment is completed, we can represent and analyze how the application delay behaves as a function of the number of concurrent clients. This behavior is shown in figure 4.15. Further, we applied the fitting toolbox provided by Matlab

Institution	Location	ID
Indiana University	Bloomington, IN	IU
Cloudlab	Salt Lake City, UT	CL
University of California, Los Angeles	Los Angeles, CA	UCLA
University of Kentucky	Lexington, KY	UKY
Naval Postgraduate School	Monterey, CA	NPS
New York University	New York, NY	NYU
University of Wisconsin	Madison, WI	UW
Clemson University	Clemson, SC	CU
MAX Giga POP	College Park, MD	MAX
NYSERNET	Syracuse, NY	NYS
Kettering University	Flint, MI	KU
University of California, Berkeley	Berkeley, CA	UCB
University of California, Davis	Davis, CA	UBD
GPO Lab	Boston, MA	GPO
Florida International University	Miami, FL	FIU

Table 4.1: List of clients used in the first latency experiment.

and obtain a regression formula:

$$AppDelay = 4.19 \times 10^{-3} N^2 - 26.44 \times 10^{-3} N + 742.34 \times 10^{-3} \tag{4.6}$$

where *AppDelay* is the application delay in seconds at the Indiana University client and N is the number of concurrent clients at the Georgia Tech server.

4.8.3.2 Second Scenario

In this case we evaluated the application delay of all clients listed in Table 4.1, for the cases where there is only one client at the time and when all clients are accessing the server at the same time. Consequently, in this case we have $N = \{1, 15\}$ and $k = 1$. Figure 4.16 shows the application delay for both situations, for each of the clients.

Additionally, based on the results of the application delay and the ping delay from each client to the servers, we can estimate how the latter affects the former. This

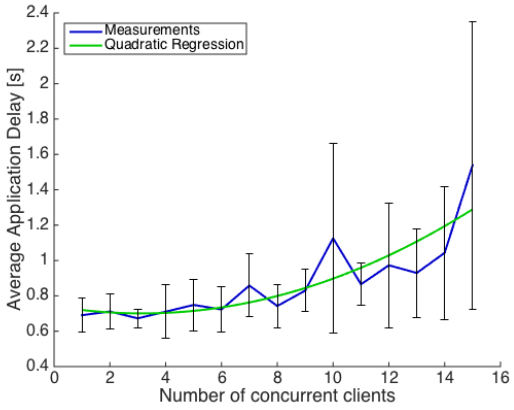

Figure 4.15: Application delay as a function of the number of concurrent clients, and its regression curve.

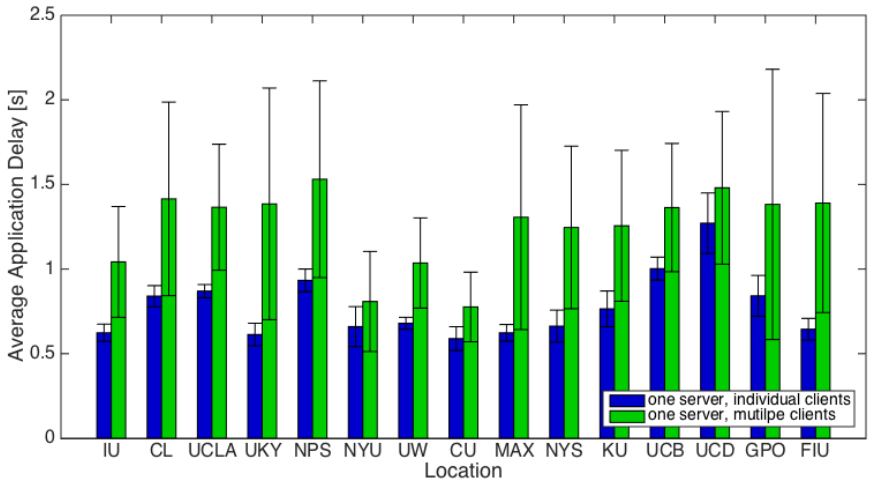

Figure 4.16: Average of the application delay for each of the clients.

behavior is presented in Figure 4.17. The left graph of the figure is the application delay as a function of the ping delay for single client. The right side, is the same scenario but for 15 concurrent clients. Moreover, we can apply a regression process and obtain formulas that describe the relationship between the application delay and ping delay for single clients and multiple clients. We present such formulas as follows:

$$AppDelay_{SingleClient} = 10.19 \times 10^{-3} PingDelay + 383.78 \times 10^{-3} \quad (4.7)$$

$$AppDelay_{MultipleClients} = 4.61 \times 10^{-3} PingDelay + 1.08 \quad (4.8)$$

where $AppDelay_{SingleClient}$ is the application delay in seconds for the single-client case, $AppDelay_{MultipleClient}$ is the application delay in seconds for the multiple-clients case, and $PingDelay$ is the ping delay in milliseconds.

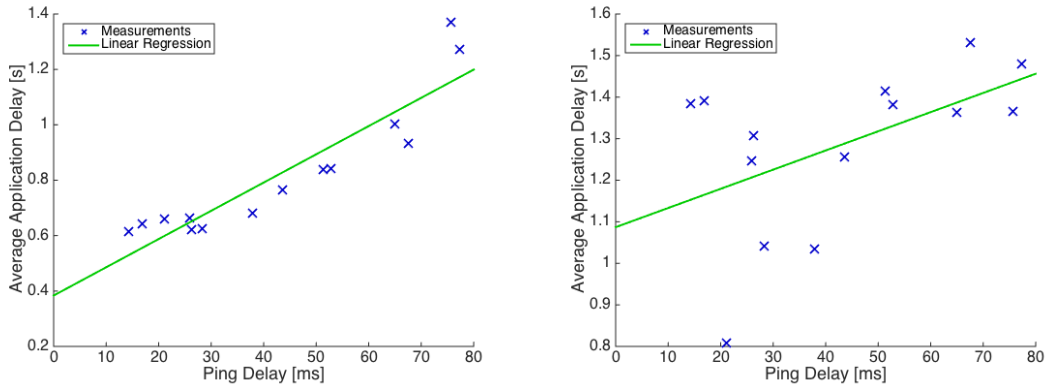

Figure 4.17: Application delay as a function of ping delay for single clients ($N = 1$) and multiple clients ($N = 15$).

In this experiment we can see that the average application delay is 774.45 milliseconds for individual clients and 1243.16 milliseconds for all the clients accessing the server concurrently. The two figures are greater than the 150 milliseconds that is expected in a real-time application. In the following sections, we show techniques to decrease this application delay.

4.8.3.3 Third Scenario

For the third experiment we added two servers to the one we already had. Consequently, the users now can select between three servers located at Georgia Tech, Kettering University and UCLA. The clients use the ping delay to decide which server to connect to. Table 4.2 provides the ping delay for each client to each server.

		Server		
		UCLA	KU	GA
Client	IU	66.40	16.57	28.22
	CL	14.62	72.58	51.31
	UCLA	*	94.09	75.64
	UKY	70.59	49.87	14.32
	NPS	10.61	102.88	67.62
	NYU	77.48	29.81	21.13
	UW	77.32	41.87	37.90
	CU	*	*	*
	MAX	71.90	24.56	26.32
	NYS	82.21	34.85	25.86
	KU	94.08	*	43.62
	UCB	47.67	56.72	65.03
	UCD	11.79	104.32	77.41
	GPO	68.03	33.03	52.76
	FIU	59.63	52.32	16.82

Table 4.2: Ping delay (in milliseconds) of each client to each server for the third scenario. The shaded cell represents the smallest ping delay. An asterisk indicates that the ping command did not received an echo reply.

Consequently, the server located in Georgia Tech provides services to CU, FIU, NYU, NYS, UKY, and UW; the server in Kettering University is accessed by the clients in IU, GPO, and MAX; the server located in UCLA provides access to CL, NPS, UCB, and UCD. In this case we will have $N_{GA} = \{1, 6\}$ for the server located at Georgia Tech, $N_{KU} = \{1, 3\}$ for the server located at Kettering University, $N_{UCLA} = \{1, 4\}$ for the server located at UCLA, and $k = 3$. Figure 4.18 shows the topology of the network. It has to be noted that in this case, Kettering University and UCLA cannot be clients and servers at the same time due to limitations in GENI's networking configuration. As a consequence, these locations were not used as clients.

After running the experiment, we observe that the application delay is reduced on average by 19.34% (from 774.45 to 624.69 milliseconds) for the case where there is only one client, and that the application delay for multiple clients is reduced on average by 44.09% (from 1243.16 to 695.07 milliseconds).

Figure 4.18: Distribution of the clients across the country.

We must remark that the fact that the Kettering Client and the UCLA client are not present in this network also contributes to decrease the application delay, as there are less concurrent clients per server. However, this effect is not significant, as the number of concurrent clients is small. For the case of the Kettering Universiy server, N is 3 instead of 4, and for the case of the UCLA server N is 4 instead of 5. Figure 4.19 shows the results the experiments.

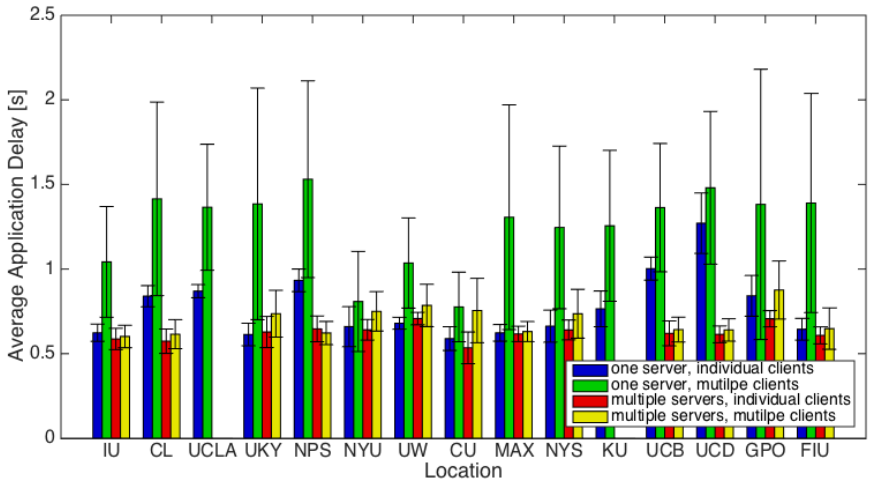

Figure 4.19: Average of the application delay for each of the clients for single server and multiple servers.

Finally, we can analyze how the application delay behaves as a function of the ping delay for each of the servers and for either single users or multiple users. For

the clients that connect to the Kettering University server, we have the following equations for application delay as a function of ping delay, for both single clients and three concurrent clients.

$$AppDelay_{SingleClient} = 7.32 \times 10^{-3} PingDelay + 455.50 \times 10^{-3} \tag{4.9}$$

$$AppDelay_{3Clients} = 16.80 \times 10^{-3} PingDelay + 287.23 \times 10^{-3} \tag{4.10}$$

Figure 4.20 provides a graphical representation of the behavior of the application delay as a function of the ping delay, for both single clients (left side of the graph) and three concurrent clients (right side of the graph).

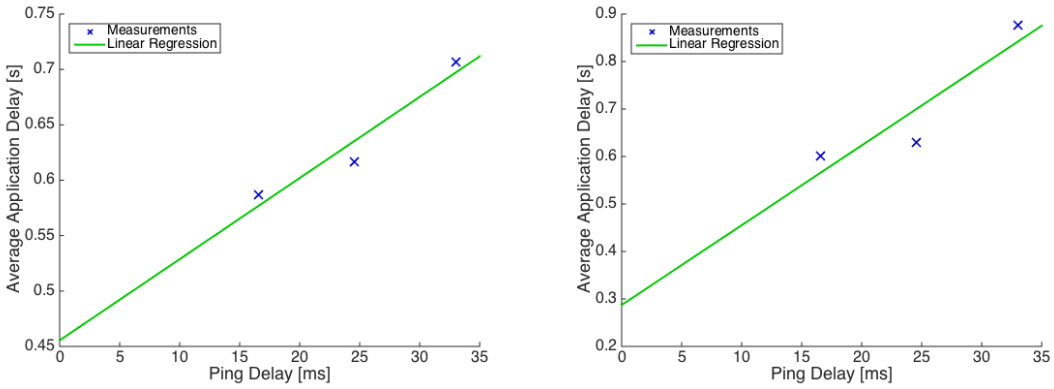

Figure 4.20: Application delay as a function of ping delay for single clients and multiple clients, for Kettering University server.

We repeat the same analysis for the rest of the servers. We compare how the application delay behaves as a function of the ping delay for both, a single client at the time and all the available clients for such server. The results of this analysis show that for the UCLA server, we have the following regression formulas:

$$AppDelay_{SingleClient} = 81.36 \times 10^{-6} PingDelay + 611.58 \times 10^{-3} \quad (4.11)$$

$$AppDelay_{4Clients} = 430.88 \times 10^{-6} PingDelay + 620.64 \times 10^{-3} \quad (4.12)$$

These results can be represented as a graph that includes the obtained valued and a linear regression based on these values. Figure 4.21 shows the results for a single client (left side of the graph) and for the four concurrent clients (right side of the graph).

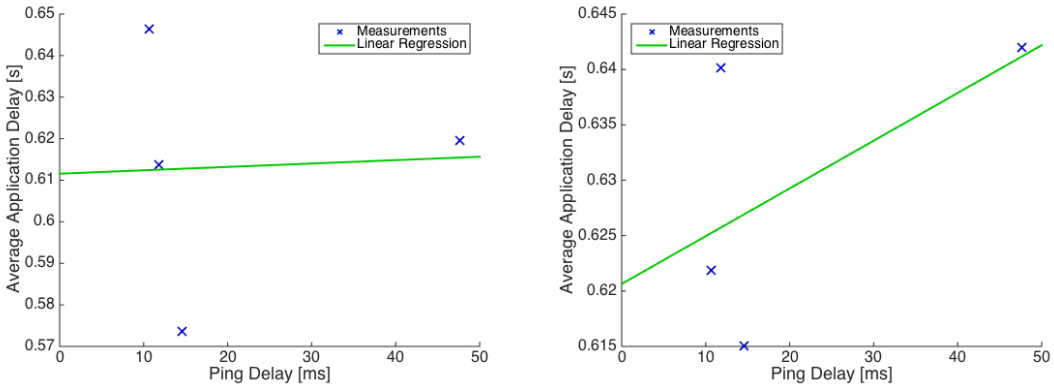

Figure 4.21: Application delay as a function of ping delay for single clients and multiple clients, for UCLA server.

Finally, we perform the same analysis for the server in Georgia Tech. First we obtain the application delay as a function of the ping delay for single clients and then for six concurrent clients. Then, we obtain the following regression equations:

$$AppDelay_{SingleClient} = 3.49 \times 10^{-3} PingDelay + 565.05 \times 10^{-3} \quad (4.13)$$

$$AppDelay_{6Clients} = 4.64 \times 10^{-3} PingDelay + 616.76 \times 10^{-3} \quad (4.14)$$

Consequently, we represent such behavior in Figure 4.22, where the left side of the graph shows the results of the experiment and the regression analysis for single clients and the right side of the graph shows the same information but for six concurrent clients.

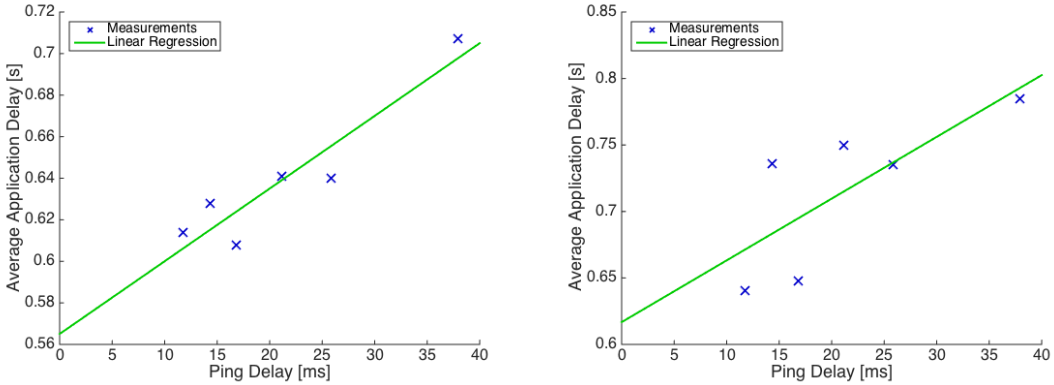

Figure 4.22: Application delay as a function of ping delay for single clients and multiple clients, for Georgia Tech server.

From this third experiment and its results we claim that the application delay not only depends on the ping delay from each client to the server, but also on the number of concurrent clients that connect to the server. First, by making the clients connect to servers that have a lower ping delay, the application delay decreases. Second, if multiple clients connect simultaneously to the same server, we see that the application delay increases compared to the case where the clients access the same server without any other client accessing the server at the same time.

4.8.3.4 Fourth Scenario

In the third scenario we showed how the application delay can be decreased when the clients connect to the server that has the minimum ping delay. However, this might lead to situations where one server has many concurrent clients, while the rest of the servers present only a few. Under this condition, the average application delay will be negatively impacted.

In this scenario we propose an algorithm that deals with this situation. The main goal behind this algorithm is to equalize the number of concurrent clients in the servers, in order to minimize the average application delay. Such algorithm, which is presented in Algorithm 1, is individually performed by each client.

Data: δ and $ServerList$
Result: $ServerName$
1 Obtain $PingDelay$ and N from each server;
2 Obtain $minPingDelay$ and corresponding $Position$ in $PingDelay$;
3 **foreach** i **in** $SeverList$ **do**
4 | **if** $(minPingDelay + \delta) > PingDelay_i$ **and** $N_i < N_{Position}$ **then**
5 | | $Position = i$;
6 | **end**
7 **end**
8 $ServerName = ServerList_{Position}$;

Algorithm 1: Proposed algorithm to reduce application delay.

The main concept behind this algorithm is simple: after obtaining the ping delay to each server, pool the servers for the number of concurrent clients (represented as N), then find the minimum ping delay, and connect to the server that has the smallest N and whose ping delay is smaller than a threshold. This threshold is set by the minimum ping delay plus a parameter δ set by the administrators.

This algorithm presents a time complexity of $O(ck)$, where c is a constant and k is the number of servers. Further, this algorithm is decentralized as each client runs it

Institution	**Location**	**ID**
Indiana University	Bloomington, IN	IU
Cloudlab	Salt Lake City, UT	CL
University of Kentucky	Lexington, KY	UKY
Naval Postgraduate School	Monterey, CA	NPS
New York University	New York, NY	NYU
University of Wisconsin	Madison, WI	UW
Clemson University	Clemson, SC	CU
MAX Giga POP	College Park, MD	MAX
NYSERNET	Syracuse, NY	NYS
University of California, Davis	Davis, CA	UBD
GPO Lab	Boston, MA	GPO
Florida International University	Miami, FL	FIU
Cornell University	Ithaca, NY	COR
University of Chicago	Chicago, IL	CHI
Stanford University	Stanford, CA	SU

Table 4.3: List of clients used in the fourth latency experiment.

independently. Additionally, we point out that in the case where $\delta = 0$, the algorithm behaves as if the clients were in the third scenario and only connect to the server with the minimum the ping delay, regardless of the number of concurrent clients.

Based on this algorithm we set up a new experiment in GENI. This time, with 15 clients ($n = 15$) and 3 servers ($k = 3$). In this case, we used the same servers located in Georgia Tech, UCLA and Kettering University, but we used a different list of clients. This list is presented in Table 4.3. Additionally, we present the ping delay from each client to each server in Table 4.4

Before running the experiment in GENI, we obtained the ping delay for each client to each of the servers. Then, we executed the algorithm for different values of δ for each of the clients. In order to do that, we run the algorithm for the first client, then for the second client and so on. Finally, after the algorithm is run for the last client, we obtain a connection matrix that indicates the server where each client connects to. For our case, we have different configurations for $\delta = \{0, 2, 4, 24\}$.

		Server		
		UCLA	KU	GA
Client	IU	66.2	19.92	80.48
	CL	14.48	67.46	108.60
	UKY	70.56	44.66	14.60
	NPS	10.58	97.92	67.36
	NYU	77.12	24.50	23.84
	UW	77.16	53.66	91.60
	CU	61.04	35.20	74.11
	MAX	71.86	19.50	26.14
	NYS	82.12	29.60	26.08
	UCD	11.64	99.24	104.00
	GPO	84.96	35.62	34.16
	FIU	60.06	47.10	16.74
	COR	83.96	31.56	27.68
	CHI	60.18	47.6736.24	46.76
	SU	79.91	97.32	102.00

Table 4.4: Ping delay (in milliseconds) of each client to each server for the fourth scenario. The shaded cell represents the minimum ping delay.

Based on these values of δ, we run the different configurations in GENI, and we obtain the application delay for each client. The results of this experiment are presented in Figure 4.23. In this figure, the minimum application delay is found for the case of $\delta = 2$. This application delay is 719.42 milliseconds, and it is 5.3% smaller than then one obtained for $\delta = 0$ (which is 759.33 milliseconds). Further, in this figure we included the number of concurrent clients that connects each of the servers.

However the arrival order of the clients conditions the results of this experiment. In order to consider this situation, we obtained an heuristic formula that outputs similar results (within a 5% margin) than the results obtained from GENI. This formula is presented as follows:

$$AppDelay_i = 10^{-3}(0.083PingDelay_{i,j}^3 - 55PingDelay_{i,j} - 100 + 18.1N_j^2 - 60N_j) \tag{4.15}$$

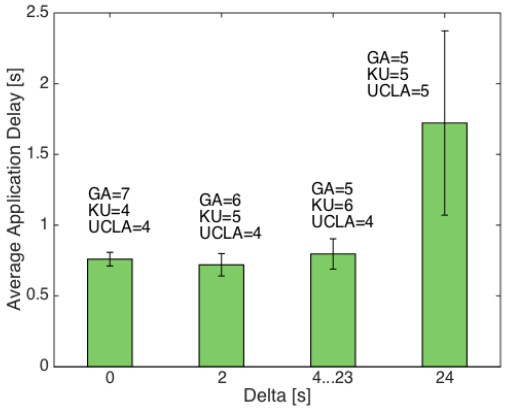

Figure 4.23: Average of the application delay for different values of δ.

where $AppDelay_i$ is the corresponding application delay for each client (i), the variable $PingDelay_{i,j}$ is the ping delay for each client (i) and each server (j), and N_j is the number of concurrent clients for each server (j).

Using Matlab we simulated the experiment for the same 15 clients, considering 10,000 random arrivals, and obtained the average of the application delay for $\delta = \{2, 4, 6, \ldots, 26\}$. We present the code that generated this simulation in Appendix C.

Further, we were interested in comparing the results of our heuristic algorithm with the optimal solution that minimizes the average application delay. In order to do that, we formulated a model for OPL/CPLEX, which is presented in Model 4.1. In this model, n is the number of clients, k is the number of servers, and cc is a Boolean matrix of n rows and k columns that represent the connectivity of the clients to the servers. Consequently, this model presents $n \times k$ variables and n constraints.

Model 4.1 is an optimization problem formulation that consists of values of cc that minimize the average application delay, with the condition that a client can connect only to one server. For a brute force approach, the time complexity of this problem is $O(k^n)$ as we need to consider all the possible combinations. Moreover, to achieve the optimal solution, we need to know the ping delays from each client before running

Variables: $cc_{i,j} \in 0, 1$
Parameters: $n, k, PingDelay_{i,j}$
Function: minimize $\frac{10^{-3}}{n} \sum_{i=1}^{n}((0.083PingDelay_{i,j}^{3} - 55PingDelay_{i,j} - 100 + 18.1N_j^2 - 60N_j) \times cc_{i,j})$
Constraints: for all i in $1..n$ $\sum_{j=1}^{k} cc_{i,j} = 1$

Model 4.1: OPL/CPLEX Model.

		Number of clients				
		15	45	75	120	150
OPL/CPLEX		0.69496	5.01110	10.85877	27.34117	43.87045
Matlab (δ)	0	0.75462	5.04814	16.36965	28.09692	56.58935
	2	0.71462	5.07378	11.33197	27.42543	46.02089
	4,6	0.72892	5.08750	11.19599	27.44796	45.93721
	8,10	0.73611	5.07150	11.20110	27.42034	45.94170
	12,14,...,20	0.77104	5.19954	11.23941	27.49544	45.99091
	22	0.73611	5.07156	11.20111	27.42039	45.94172
	24	0.77104	5.19954	11.23941	27.49544	45.99091
	26	1.55639	6.43857	11.21816	27.91556	44.19000

Table 4.5: Average application delay in seconds provided by the OPL/CPLEX and the Matlab simulations. The shaded cells indicate the smallest application delay obtained from the Matlab Simulation.

the model. The code used for this implementation in OPL/CPLEX is presented in Appendix C.

Finally, we considered the cases where there are more than 15 clients and simulated scenarios for 45, 75, 120 and 150 clients and 3 servers. In order to obtain this numbers of clients, we randomly selected them from the original list of 15, obtaining the corresponding ping delay matrix for each of the cases. Consequently, we simulated these scenarios in both Matlab and OPL/CPLEX. Given the high number of possible combinations, we run our OPL/CPLEX model for 8 hours in the case of 45 and 75 users, and 12 hours in the case of 120 and 150 users, and report the best solution obtained so far. We present our results in Table 4.5.

Based on the results obtained from Table 4.5, we can compare the results of our proposed algorithm and the solution found by OPL/CPLEX. We conclude that the best solution that our algorithm can find (considering all the values of δ) is at most 3.11% bigger than the optimal solution found by OPL/CPLEX.

The application delay results we obtained are greater than the 150 milliseconds limit. However, we can decrease the application delay by changing the encoding of the images the server sends to the client. In our experiments we have used raw encoding because of compatibility limitations in the client software. However, a comparison with other VNC clients shows that the application delay can be reduced up to a 80.0% by using hextile encoding, which decreases the size of the data to be transmitted from the server to the client. The implementation of this encoding would reduce the latency on average to 124.94 milliseconds for individual users and to 139.01 milliseconds for 13 concurrent users, in the case of the third scenario. For the fourth scenario, the average application delay for 15 concurrent clients would be reduced to 142.95 milliseconds if we apply our algorithm. Further, there are other encoding techniques, such as VNC Tight Encoder, that claim even better encoding performance [152].

In this chapter, we described and analyzed in detail the performance of RMS. We described an implementation and showed experimental results related to this implementation. In the next chapter, we will provide our conclusion and future research directions related to RMS.

Chapter 5

Conclusion and Future Work

This chapter presents the conclusion of this thesis and directions for future research.

5.1 Conclusions

The introduction of BYOD policies are beneficial for both the enterprise and its employees, as it increases job satisfaction, the employees become more productive, the enterprise can use it to recruit potential employees, while it reduces costs related to assets and operation.

However, BYOD policies and the mobility nature of BYOD devices pose security threats to the enterprise information, as well as the employee privacy. This creates the challenges of data leakage, unauthorized sharing of spaces, lack of security compliance, and employee privacy.

In order to address these challenges, a secure BYOD environment must meet the goals of space isolation, corporate data protection, security policy enforcement, true space isolation, non-intrusiveness, and low resource consumption.

A classification of the current solutions for BYOD has been presented. We can

categorize the solutions into Mobile Virtual Machine, Agent-based, Cloud-based, Virtual Private Network, Trusted Environments, and Framework. We summarize which goals they meet, and we show that currently there is no solution that achieves all of these goals.

In this thesis we proposed a new framework for a secure BYOD environment, Remote Mobile Screen (RMS), which meets all the necessary goals for a secure BYOD environment. Our framework mainly consists on deploying a personal space on the mobile device, and deploying a corporate space at the corporate network. Then, the employee accesses his or her corporate space through the use of a VNC client.

We describe the architecture of RMS, described how it works, discussed its features, and showed an example of an implementation using commonly available software. Further, we provide a security analysis on our architecture, and discussed the challenges related to RMS. We have showed results of our experiments related to compatibility, scalability and latency.

In our compatibility experiment, we showed that at least 90.2% of the productivity applications found in Google Play can be installed in mobile OSes for the x86 architecture, and that 80.4% of the applications work on these mobile OSes.

In our scalability experiment, we tested our implementation on a high performance server provided by HCC. In this experiment, we showed that up to 596 VMs can be run in such server. We concluded that the RAM requested by the VM is not a factor in determining of how many concurrent VMs we can support. Further, by limiting the RAM available, we observed that the number of VMs is linearly related to the amount of available RAM, where 2.3 VMs can be run for each GB of available RAM.

Finally, in our latency experiment we showed that the application latency experienced in a particular client increases as a function of the number of concurrent clients accessing a server, and the ping delay from the client to the server. Further we showed

how the application delay can be decreased below an acceptable value of 150 milliseconds by deploying multiple servers close to the clients and by using high-compression encoding formats in the VNC protocol.

5.2 Directions for Future Work

We present a series of directions for future research. We would like to develop a Corporate Space Manager. This implementation must consider the centralized and distributed scenarios that different enterprises may present. We plan to improve our implementation of RMS by including all the components that were not used in our proof-of-concept. Another research directions involve the possibility of having multiple users in a single implementation of Android, in order to increase the scalability performance. Further, we would like to explore the implementation of iOS on virtual machines, either for x86 or ARM architectures. Future work also includes developing algorithms and solutions for content distribution in distributed networks where users modify the content. Since the VNC protocol might not be suitable for mobile applications, a new communication protocol could be developed. This protocol must consider the gestures and features proper of mobile devices, as well as the requirements for a secure communication. Further, the introduction of a new communication protocol will include developing client and server applications for different mobile platforms. Other research direction involves implementing other techniques for reducing the latency in the VNC protocol. Finally, we plan to perform usability surveys that consider the feedback from real users.

The introduction of BYOD policies, which allow employees to use their personal mobile devices, create serious security challenges for both the enterprise and the employees. These challenges must not be ignored by the enterprises and proper

solutions must be implemented in order to protect both the corporate data as well as the personal data.

Appendix A

List and Description of Mobile Applications Tested in the Usability Experiment

Below we present a set of tables that contain the name of the mobile application used in the experiment in section 4.7.3, as well as a short description of the type of each application. Given the number of applications tested, we show the information in two tables.

Table A.1: List of mobile applications analyzed.

Application	**Description**
Advanced Task Manager - Boost	Task Manager
ai.type keyboard Plus + Emoji	Keyboard
Application	Description
AirDroid - Android on Computer	Remote Notification
AntiVirus Security - FREE	Antivirus
Basecamp - Official App	Project Management
Cisco WebEx Meetings	Web Conference
Cloud Print	Printing
ColorNote Notepad Notes	Note Taking
Dropbox	Cloud Storage
Easy Uninstaller App Uninstall	Uninstaller
Echo Notification Lockscreen	UI Enhancer
Evernote	Note Taking
EverythingMe Launcher	UI Enhancer
Flipboard: Your News Magazine	News Reader
Gmail	Email
Google Docs	Word Processor
Google Drive	Cloud Storage
Google Keep - notes and lists	Note Taking
Google Sheets	Spreadsheets
Google Slides	Presentations
Google Text-to-Speech	UI Enhancer

Table A.2: List of mobile applications analyzed (contd.).

Application	**Description**
IF by IFTTT	Automation
Link Bubble Browser	Web Browser
Application	Description
Mailbox	Email
Microsoft Office for Mobile	Productivy Suite
Microsoft OneDrive	Cloud Storage
Microsoft OneNote	Note Taking
Microsoft Outlook Preview	Email
Papyrus - Natural notetaking	Note Taking
Pocket	News Reader
Quip: Docs, Chat, Spreadsheets	Productivy Suite
Skitch - Snap. Mark up. Send.	Note Taking
Slack	Communication
Splashtop Remote Desktop HD	Remote Access
Stitcher Radio for Podcast Manager	Podscast
Sunrise Calendar	Callendar
SwiftKey Keyboard + Emoji	Keyboard
TeamViewer for Remote Control	Remote Access
Todoist: To-Do List, Task List	To-Do List
WPS Office + PDF	Productivy Suite
Wunderlist: To-Do List & Tasks	To-Do List

Appendix B

Scripts Used in the Scalability Test

Below we show the scripts used to perform the experiments related to the scalability test presented in section 4.8.2.

Listing B.1: run_experiment.sh

```
#!/bin/bash

# This scripts turns the VM in a server and obtains statistics.

# We that the arguments of the command are correct. The name of an
    output file is needed.
if [ "$#" -ne 1 ]; then
    echo "Usage" $0 "OUTPUT-FILE"
    exit
fi

# We define a function that obtains the desired statistics.
function getstats {
    MEMORY=`free -t -m | grep "Mem:\|cache:\|Swap:\|Total" | awk '{
        mem=mem","$3} END {print mem}'`
    CPU=`top -b -n2 | grep "Cpu(s)"|tail -n 1 | awk '{print $2 + $4
        }'`
    THREADS=`ps -eo nlwp | tail -n +2 | awk '{ num_threads += $1 }
        END { print num_threads }'`
    LOAD=`top -b -n1 | grep "load" | awk '{print $11 $12 $13}'`
    echo $MEMORY","$CPU","$THREADS","$LOAD
}

# We start the expriment, obtain the first statistics and record
    them in an output file.
```

```
21 echo "Starting Experiment" >> $1
22 date "+%D %T.%N" >> $1
23 getstats >> $1
24 echo >> $1
25
26 # The for loop used to issue the commands a pre-defined number of
       times.
27 for i in `seq 1 768`;
28 do
29    #  We obtain information about the VM.
30    HOST=`VBoxManage guestproperty enumerate androVM_$i | grep
          androvm_ip_management | awk '{ print substr($4, 0, length(
          $4))}'`
31    VM="androVM_"$i
32    START=`date "+%T.%N"`
33    COUNTER=0
34    UNREACHABLE=1
35
36    # We start the VM.
37    VBoxManage startvm $VM --type headless
38
39     # Repeat this while the VM is still booting.
40     while [ "$COUNTER" -lt 180 -a "$UNREACHABLE" -ne 0 ] ; do
41
42       # If the host has not an IP assigned yet, wait a second.
43       if [[ $HOST == *"10.10.0.100"* ]] ; then
44          sleep 1;
45          HOST=`VBoxManage guestproperty enumerate androVM_$i |
              grep androvm_ip_management | awk '{ print substr($4,
              0, length($4))}'`
46
47       # Else, ping the VM.
48       else
49          ping -q -c1 -w 1 $HOST && UNREACHABLE=$?
50       fi
51
52       # Count the number of times we tried to reach the VM.
53       COUNTER=$((COUNTER + 1))
54    done
55
56    # If the VM is unreachable, print a message and exit the script
          .
57    if [ "$UNREACHABLE" -ne 0 ] ; then
58       echo "$HOST is unreachable"
59       exit
60
61    # Else, record stats of the VM, stop it and restart it.
62    else
63       REACHED=`date "+%T.%N"`
64       STATS=`getstats`
65       VBoxManage controlvm $VM savestate
```

```
66        PAUSED=‘date "+%T.%N"‘
67        VBoxManage startvm $VM --type headless
68        RESTARTED=‘date "+%T.%N"‘
69        echo $VM","$HOST$STATS","$START","$REACHED","$PAUSED","
              $RESTARTED >> $1
70    fi
71 done
72
73 # Finally, record final messages in the output file.
74 echo >> $1
75 echo "Experiment finished" >> $1
76 date "+%D %T.%N" >> $1
77 getstats >> $1
```

Listing B.2: stop_experiment.sh

```
1  #!/bin/bash
2
3  # This script stops all the VMs.
4  # The for loop is used to execute the programs a pre-defined
       number of times.
5  for i in ‘seq 1 512‘;
6  do
7     echo "Powering off androVM_"$i
8
9     # Stop VM in VirtualBox
10    VBoxManage controlvm androVM_$i poweroff
11
12    # If there is an error in the previous command, there are no
          VMs to stop, and we exit the script.
13    if [ $? -eq 1 ]
14    then
15      sleep 10
16      ps -aux | grep VBox
17      exit
18    fi
19
20    # Obtain the list of VMs still running and kill them
21    PROCESSES=‘ps -aux | grep VBoxHeadless | awk ’{PID=PID" "$2}
          END {print PID}’‘
22    kill -9 $PROCESSES
23 done
```

Listing B.3: create_vms.sh

```
1  #!/bin/bash
2
3  # This scripts creates virtual machines in VirtualBox based on an
       OVA file.
4  # The for loop used to issue the command a pre-defined number of
       times.
5
```

```
6 for i in `seq 1 768`;
7 do
8    VBoxManage import ./androVM_benchmark.ova --vsys 0 --vmname
        androVM_$i
9 done
```

Listing B.4: delete_vms.sh

```
1 #!/bin/bash
2
3 # This scripts deletes virtual machines in VirtualBox based on the
      VM name.
4 # The for loop used to issue the command a pre-defined number of
     times.
5
6 for i in `seq 1 768`;
7 do
8    VBoxManage unregistervm androVM_$i --delete
9 done
```

Appendix C

Scripts Used in the Latency Test

Below we show the scripts used to perform the experiments related to the latency test presented in section 4.8.3.

Listing C.1: client_fedora_script.sh

```
1 #!/bin/bash
2
3 # This script installs all the needed software for setting up a
      client running Fedora in GENI.
4
5 # Install modify the repositories, install python-pip, PIL and
      vncdotool.
6 sudo sed -i 's/https/http/g' /etc/yum.repos.d/fedora-updates.repo;
       sudo sed -i 's/https/http/g' /etc/yum.repos.d/fedora.repo;
      sudo yum -y install python-pip; sudo pip-python install PIL;
      sudo pip-python install vncdotool
7
8 # Obtain the ping delay to each server and record them in output
      files.
9 ping -c 5 143.215.216.146 >> ping_GA.txt;
10 ping -c 5 164.67.126.19 >> ping_UCLA.txt;
11 ping -c 5 192.245.254.18 >> ping_KET.txt
12
13 # Obtain the route to each server and record them in output files.
14 traceroute 143.215.216.146 >> traceroute_GA.txt
15 traceroute 164.67.126.19 >> traceroute_UCLA.txt
16 traceroute 192.245.254.18 >> traceroute_KET.txt
```

Listing C.2: client_ubuntu_script.sh

```
1 #!/bin/bash
```

```
2
3 # This script installs all the needed software for setting up a
      client running Ubuntu in GENI.
4
5 # Install modify the repositories, install python-pip, PIL and
      vncdotool.
6 sudo apt-get update; sudo apt-get -y install python-dev python-pip
      ; sudo pip install --allow-unverified PIL vncdotool
7
8 # Obtain the ping delay to each server and record them in output
      files.
9 ping -c 5 143.215.216.146 >> ping_GA.txt
10 ping -c 5 164.67.126.19 >> ping_UCLA.txt
11 ping -c 5 192.245.254.18 >> ping_KET.txt
12
13 # Obtain the route to each server and record them in output files.
14 traceroute 143.215.216.146 >> traceroute_GA.txt
15 traceroute 164.67.126.19 >> traceroute_UCLA.txt
16 traceroute 192.245.254.18 >> traceroute_KET.txt
```

Listing C.3: server_script.sh

```
1 #!/bin/bash
2
3 # This script installs all the needed software for setting up a
      server running Ubuntu in GENI.
4
5 # Mount an additional hard drive to have extra storage for the VMs
      .
6 /usr/testbed/bin/mkextrafs /mnt
7
8 # Configure the repositories and install VirtualBox 4.3
9 echo "deb http://download.virtualbox.org/virtualbox/debian precise
       contrib" | sudo tee -a /etc/apt/sources.list
10 wget -q http://download.virtualbox.org/virtualbox/debian/
      oracle_vbox.asc -O- | sudo apt-key add -
11 sudo apt-get update
12 sudo apt-get install virtualbox-4.3
13
14 # Configure the repositories and install adb.
15 sudo add-apt-repository ppa:nilarimogard/webupd8
16 sudo apt-get update
17 sudo apt-get install android-tools-adb
18
19 # For loop to create a predefined number of VMs (provided as the
      first argument).
20 for i in `seq 1 $1`;
21 do
22
23     # Import a VM in VirtualBox using an OVA file. We need to
          provide the path to the mounted hard drive.
```

```
24    VBoxManage import ./androVM.ova --vsys 0 --vmname androVM_$i --
          vsys  0 --unit 10 --disk /mnt/androVM_$i/androVM-disk1.vmdk
           --unit 11 --disk /mnt/androVM_$i/androVM-disk2.vmdk --unit
           12 --disk /mnt/androVM_$i/androVM-disk3.vmdk
25 done
26
27 # Create a hostonly interface. This is used by AndroVM for
       management.
28 VBoxManage hostonlyif create
29
30 # Create a NAT network with a DHCP
31 VBoxManage natnetwork add --netname "NatNetwork" --network
       10.0.2.0/24  --enable --dhcp on
32
33 # For loop to create a predefined number of por-forwarding rules (
       provided as the first argument).
34 for i in `seq 1 $1`;
35 do
36
37    # For each VM, add a forwarding rule.
38    VBoxManage natnetwork modify --netname NatNetwork --port-
          forward-4 "guestvnc$i:tcp:[143.215.216.146]:5090$i:[10.0.2.
          $(i+3)]:5901"
39 done
```

Listing C.4: model.m

```
1 % This Matlab Scripts simulates how a GENI implementation behaves
2 % based on the heuristic we proposed. It randomly selects n users
3 % from a set of 15 users, it randomly decides 10,000 arrival
      orders,
4 % and obtains the average application delayfor each oder, for
5 % different values of delta. The output file includes each of
      these
6 % average application delays and the ping delay matrix for the n
7 % clients selected.
8
9 % Clear the variables and the console.
10 clear
11 clc
12
13 % Select the name of the output file.
14 fileID = fopen('exp.txt','w');
15
16 % Parameters.
17 delta=0:2:26;
18 n=150;
19 k=3;
20
21 % Original ping delay for the 15 clients in GENI.
22 orig_pingdelay = [
```

```
66.2    19.92   80.48
14.48   67.46   108.6
70.56   44.66   14.6
10.58   97.92   67.36
77.12   24.5    23.84
77.16   53.66   91.6
61.04   35.2    4.112
71.86   19.5    26.14
82.12   29.6    26.08
11.64   99.24   104
84.96   35.62   34.16
60.06   47.1    16.74
83.96   31.56   27.68
60.18   36.24   46.76
9.916   97.32   102
];

% Create ping delay matrix for n clients and k servers and
% randomly select the clients from the original matrix.
pingdelay=zeros(n,k);

for i=1:n
    pingdelay(i,:)=orig_pingdelay(randi(15),:);
end

% In case we only consider the 15 original clients from GENI.
%  pingdelay=orig_pingdelay;

% Repeat 10.000 times.
for l=1:10000

        % Obtain the size of delta and create the arrival matrix.
    [dn dk]=size(delta);
    arrival=randperm(n);

        % Loop for each of the elements in delta
    for m=1:dk

        % Create the connectivity matrix and the vector that
        % contains the number of concurrent clients for each
           server.
        cc=zeros(n,k);
        N=zeros(k,1);

        % Loop for each of the n clients.
        for i=1:n

             % Obtain the minimum delay and its position in the
             % matrix.
            [minimum,pos]=min(pingdelay(arrival(i),:));

```

```
73              % Check if any ping delay satisfies the conditions in
74              % the algorithm.
75              for j=1:k
76                  if ((minimum + delta(m)) > pingdelay(arrival(i),j)
                        && (N(j)<N(pos)));
77                      pos=j;
78                  end
79              end
80
81              % Update the cc matrix and the N vector.
82              cc(arrival(i),pos)=1;
83              N(pos)=N(pos)+1;
84          end
85
86          % Initialize the vector for computing the application
87          % delay
88          appdelay=zeros(15,1);
89
90          % For each of the cells in cc
91          for i=1:n
92              for j=1:k
93
94                  % If the cell is equal to 1, compute the
95                  % application delay for the client i.
96                  if cc(i,j) == 1
97                      appdelay(i)=10^(-3)*(0.083*pingdelay(i,j)
                            ^3-...
98                          55*pingdelay(i,j)-...
99                          100+18.1*N(j)^2-60*N(j));
100                 end
101             end
102         end
103
104         % Print the results fot this of this run in the output
105         % file
106         formatSpec = 'n= %d delta= %d and meandelay= %f\n';
107         fprintf(fileID,formatSpec,n,delta(m),mean(appdelay));
108     end
109 end
110
111 % Print the ping delay matrix in the output file, so it can be
112 % used in OPL/CPLEX
113 fprintf(fileID,'PingDelay = [ \n');
114 for i=1:n
115     fprintf(fileID,'[%.2f %.2f %.2f],\n',pingdelay(i,:));
116 end
117 fprintf(fileID,'\n];');
```

Listing C.5: model1.mod

```
1 /*********************************************
```

```
 2  * OPL 12.6.0.0 Model
 3  * Author: santiago
 4  * Creation Date: Mar 24, 2015 at 3:45:30 PM
 5  *********************************************/
 6
 7 using CP;
 8
 9 execute {
10     cp.param.timeLimit=3600;
11     cp.param.OptimalityTolerance=1E-2;
12     cp.param.RelativeOptimalityTolerance=1E-1;
13 }
14
15
16 //... Arguments
17 int n = 15;
18 int k = 3;
19 float PingDelay[1..n][1..k] = ...;
20
21 //... Variables
22 dvar boolean cc[1..n][1..k];
23
24 //... Additional variables
25 range r=1..n;
26
27 //... Function
28 minimize (10^(-3)/n) * (
29    sum(i in 1..n, j in 1..k) (
30    0.083 * PingDelay[i][j]^3 -
31    55 * PingDelay[i][j] -
32    100 +
33    18.1 * (sum(r in 1..n)cc[r][j])^2 -
34    60 * (sum(r in 1..n)cc[r][j])
35    ) * cc[i][j]
36    );
37
38 //... Constraints
39 subject to{
40
41 forall(i in 1..n, j in 1..k){
42   sum(c in 1..k) cc[i][c] == 1;
43 }
44 }
45
46 //... Store the CC Matrix in an Output File
47 execute {
48    var ofile = new IloOplOutputFile("Output.txt");
49    ofile.writeln("cc = [");
50    for (var i in thisOplModel.r){
51       ofile.writeln(thisOplModel.cc[1][1]+" "+thisOplModel.cc[i
          ][2]+" "+thisOplModel.cc[i][3])
```

```
52      }
53      ofile.writeln("]");
54      ofile.close();
55  }
```

Bibliography

[1] "Cisco visual networking index: Global mobile data traffic forecast update, 2013-2018," White Paper, Cisco, Feb. 2014. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.pdf

[2] "The world population situation in 2014," Department of Economic and Social Affairs Population Division, 2014. [Online]. Available: http://www.un.org/en/development/desa/population/publications/pdf/trends/Concise%20Report%20on%20the%20World%20Population%20Situation%202014/en.pdf

[3] "BYOD: From company-issued to employee-owned devices," McKinsey & Company, June 2012. [Online]. Available: http://www.mckinsey.com/textasciitilde/media/mckinsey/dotcom/client_service/high%20tech/pdfs/byod_means_so_long_to_company-issued_devices_march_2012.ashx

[4] M. Silic and A. Back, "Factors impacting information governance in the mobile device dual-use context," *Records Management Journal*, vol. 23, no. 2, pp. 73–89, 2013.

[5] J. Bradley, J. Loucks, J. Macaulay, R. Medcalf, and L. Buckalew, "BYOD: A global perspective. Harnessing employee-led innovation," 2012. [Online]. Available: http://www.cisco.com/web/about/ac79/docs/re/BYOD_Horizons-Global.pdf

[6] D. Assing and S. Cal'e, *Mobile access safety: Beyond BYOD*. Hoboken,NJ: John Wiley & Sons, 2013.

[7] M. Loose, A. Weeger, and H. Gewald, "BYOD - The next big thing in recruiting? Examining the determinants of BYOD service adoption behavior from the perspective of future employees," in *Proc. 19th Americas Conf. Information Systems*, Chicago, IL, Aug. 2013.

[8] J. Hayes, "The device divide," *Engineering & Technology*, vol. 7, no. 9, pp. 76–78, Oct. 2012.

[9] S. Gimenez Ocano, B. Ramamurthy, and Y. Wang, "Remote Mobile Screen (RMS): an approach for secure BYOD environments," in *Computing, Networking and Communications (ICNC), Int. Conf.*, Anaheim, CA, Feb. 2015.

[10] ——, "Security challenges in and solutions for the Bring Your Own Device (BYOD) environment: a survey," 2015, under review.

[11] ——, "Implementation challenges of Remote Mobile Screen (RMS) for secure BYOD environments," 2015, under review.

[12] Y. Wang, K. Streff, and S. Raman, "Smartphone security challenges," *Computer*, vol. 45, no. 12, pp. 52–58, Dec. 2012.

[13] "Mobile threat report, July - September 2013," F-Secure Corporation, 2013. [Online]. Available: http://www.f-secure.com/documents/996508/1030743/Mobile_Threat_Report_Q3_2013.pdf

[14] "Mobile threat report, Q 2014," F-Secure Corporation, 2014. [Online]. Available: http://www.f-secure.com/documents/996508/1030743/Mobile_Threat_Report_Q1_2014.pdf

[15] Y. Zhou, Z. Wang, W. Zhou, and X. Jiang, "Hey, you, get off of my market: Detecting malicious apps in official and alternative Android markets," in *Proc. 19th Annual Network & Distributed System Security Symp.*, San Diego, CA, Feb. 2012.

[16] Google Inc. Launch checklist. [Online]. Available: http://developer.android.com/distribute/tools/launch-checklist.html

[17] E. Kim. Creating better user experiences on google play. Google Inc. [Online]. Available: http://android-developers.blogspot.ca/2015/03/creating-better-user-experiences-on.html

[18] Trend-Micro, "Mobile monthly report - Mobile phishing: A problem on the horizon," February 2013. [Online]. Available: http://about-threats.trendmicro.com/us/mobilehub/mobilereview/rpt-monthly-mobile-review-201302-mobile-phishing-a-problem-on-the-horizon.pdf

[19] "Secure our smartphone initiative, one year later," The Office of the New York State Attorney General, June 2014. [Online]. Available: http://www.ag.ny.gov/pdfs/SOS%201%20YEAR%20REPORT.pdf

[20] S. Wright, "The Symantec smartphone honey stick project," 2012.

[21] Second hand mobiles contain personal data. CPP. [Online]. Available: http://www.cppgroupplc.com/media-centre/press-releases/

[22] J. McCcolgan. Tens of thousands of Americans sell themselves online every day. AVAST Software a.s. [Online]. Available: http://blog.avast.com/2014/07/08/tens-of-thousands-of-americans-sell-themselves-online-every-day/

[23] Y. Wang, J. Wei, and K. Vangury, "Bring Your Own Device security issues and challenges," in *11th Annual IEEE Consumer Communications & Networking Conf.*, Las Vegas, NV, Jan. 2014.

[24] B. Bergstein. IBM faces the perils of "Bring Your Own Device". MIT Technology Review. [Online]. Available: http://www.technologyreview.com/news/427790/ibm-faces-the-perils-of-bring-your-own-device/

[25] M. B. Salem, S. Hershkop, and S. J. Stolfo, "A survey of insider attack detection research," in *Insider Attack and Cyber Security.* New York, NY: Springer US, 2008, vol. 39, pp. 69–90.

[26] K. W. Miller, J. Voas, and G. F. Hurlburt, "BYOD: security and privacy considerations," *It Professional*, vol. 14, no. 5, pp. 53–55, Sep. 2012.

[27] "iOS software license agreement," Apple Inc., January 2015. [Online]. Available: http://images.apple.com/legal/sla/docs/iOS81.pdf

[28] J. Andrus, C. Dall, A. V. Hof, O. Laadan, and J. Nieh, "Cells: a virtual mobile smartphone architecture," in *Proc. 23th ACM Symp. Operating Systems Principles*, Cascais, Portugal, Oct. 2011.

[29] K. Barr, P. Bungale, S. Deasy, V. Gyuris, P. Hung, C. Newell, H. Tuch, and B. Zoppis, "The VMware mobile virtualization platform: is that a hypervisor in your pocket?" *ACM SIGOPS Operating Systems Review*, vol. 44, no. 4, pp. 124–135, Dec. 2010.

[30] "Virtualization is coming to a platform near you," White Paper, ARM, Jan. 2011. [Online]. Available: http://www.arm.com/files/pdf/system-mmu-whitepaper-v8.0.pdf

[31] "Integrity multivisor - virtualization archiitecture for secure systems," White Paper, Green Hill Software, Feb. 2014. [Online]. Available: http://www.ghs.com/download/datasheets/multivisor_tmd.pdf

[32] "ARM security technology," White Paper, ARM, Sep. 2009. [Online]. Available: http://infocenter.arm.com/help/topic/com.arm.doc.prd29-genc-009492c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf

[33] VMware Horizon Mobile 1.3.1 release notes. VMWare. [Online]. Available: https://www.vmware.com/support/horizon_mobile/doc/horizon-mobile-131-release-notes.html

[34] Verizon rolls out dual persona solution with VMware Horizon Mobile to Android-based smartphones. VMWare. [Online]. Available: http://www.vmware.com/company/news/releases/vmw-horizon-Verizon-051513

[35] Horizon Suite end of availability. VMWare. [Online]. Available: http://www.vmware.com/products/horizon-suite

[36] M. Lange, S. Liebergeld, A. Lackorzynski, A. Warg, and M. Peter, "L4Android: a generic operating system framework for secure smartphones," in *Proc. 1st ACM workshop on Security and privacy in smartphones and mobile devices*, Chicago, IL, Oct. 2011.

[37] N. Leavitt, "Today's mobile security requires a new approach," *Computer*, vol. 46, no. 11, pp. 16–19, Nov. 2013.

[38] M. Eslahi, M. V. Naseri, H. Hashim, N. Tahir, and E. H. M. Saad, "BYOD: Current state and security challenges," in *Computer Applications and Industrial Electronics (ISCAIE), IEEE Symp.*, Penang, Malaysia, Apr. 2014.

[39] A. Scarfo, "New security perspectives around BYOD," in *Proc. 7th Int. Conf. Broadband, Wireless Computing, Communication and Applications*, Victoria, BC, Canada, Nov. 2012.

[40] "Mobile device management," Airwatch by VMWare, 2014. [Online]. Available: http://www.air-watch.com/downloads/brochures/AirWatch_brochure_mobile_device_management.pdf

[41] BYOD security, policy compliance for Android and iOS devices - Amtel. Amtel. [Online]. Available: https://www.amtelnet.com/solutions/mobile-security/byod-security/

[42] iPad Management & iPhone Management. Amtel. [Online]. Available: https://www.amtelnet.com/solutions/mobile-security/ipad-management-and-iphone-management/

[43] Android Device Management. Amtel. [Online]. Available: https://www.amtelnet.com/solutions/mobile-security/android-device-management/

[44] "Product overview - BlackBerry Enterprise Service 12," BlackBerry, November 2014. [Online]. Available: http://docs.blackberry.com/en/admin/deliverables/67267/Product_Overview_BES12_v12.0_en.pdf

[45] "CA mobile device management," Datasheet, CA, Sep. 2014. [Online]. Available: http://www.ca.com/us/~/media/Files/DataSheets/ca-mobile-device-management.PDF

[46] "Citrix XenMobile technology overview," Whitepaper, Citrix, Aug. 2013. [Online]. Available: http://www.citrix.com/content/dam/citrix/en_us/documents/products-solutions/citrix-xenmobile-technology-overview.pdf

[47] "Reference architecture for mobile device and app management," Citrix, 2013. [Online]. Available: http://www.citrix.com/content/dam/citrix/en_us/documents/products-solutions/reference-architecture-for-mobile-device-and-app-management.pdf

[48] "Mobile management," Datasheet, Dell, 2014. [Online]. Available: http://software.dell.com/documents/dell-mobile-management-datasheet-datasheet-68961.pdf

[49] "Mobile device management," Datasheet, Good Technology, Jul. 2014. [Online]. Available: http://media.good.com/documents/ds-mdm.pdf

[50] MDM. Good Technology. [Online]. Available: http://www1.good.com/secure-mobility-solution/mobile-device-management.html

[51] "Maas360 mobile device management," Datasheet, MaaS360, Feb. 2014. [Online]. Available: http://content.maas360.com/www/content/ds/ds_maas360_mdm_overview.pdf

[52] Mobile device management. MaaS360. [Online]. Available: http://www.maas360.com/products/mobile-device-management/?s=home

[53] "McAfee Enterprise Mobility Management (McAfee EMM) 12.0 - frequently asked questions," McAfee, 2014. [Online]. Available: http://www.mcafee.com/us/resources/faqs/faq-emm.pdf

[54] "Microsoft Intune," Datasheet, Microsoft, Oct. 2014. [Online]. Available: http://download.microsoft.com/download/4/B/F/4BF2842C-2B15-44E5-87B4-2C2949A81DE9/Microsoft_Intune_datasheet.pdf

[55] Microsoft Intune. Microsoft. [Online]. Available: http://www.microsoft.com/en-us/server-cloud/products/microsoft-intune/default.aspx

[56] "Mobileiron product packaging," Datasheet, MobileIron, Oct. 2014. [Online]. Available: https://www.mobileiron.com/sites/default/files/datasheets/files/MobileIron%202014%20Bundle%20Datasheet_V2-0_EN.pdf

[57] MobileIron Core is a key software component in the MobileIron platform. MobileIron. [Online]. Available: https://www.mobileiron.com/en/products/platform-architecture/mobileiron-core

[58] Mobile device management. MobileIron. [Online]. Available: https://www.mobileiron.com/en/solutions/mobile-device-management-mdm

[59] Securing enterprise mobility for greater competitive advantage. SAP. [Online]. Available: http://www.sap.com/bin/sapcom/en_us/downloadasset.2012-08-aug-01-18.managing-enterprise-mobility-for-greater-competitive-advantage-pdf.bypassReg.html

[60] SAP Afaria, cloud edition. SAP. [Online]. Available: http://www.sap.com/bin/sapcom/en_us/downloadasset.2013-05-may-14-18.sap-afaria-cloud-edition--a-by-the-numbers-approach-to-security-pdf.bypassReg.html

[61] "Unified enterprise mobility management," SOTI, 2015. [Online]. Available: https://www.soti.net/media/179035/SOTI_MobiControl_Brochure_redesign.pdf

[62] "Symantec Mobility: Device management," Datasheet, Symantec, Feb. 2014. [Online]. Available: http://www.symantec.com/content/en/us/enterprise/fact_sheets/b-mobility-device-management-ds-21338200.pdf

[63] D. Tynan. How Box.com allowed a complete stranger to delete all my files. IT World. [Online]. Available: http://www.itworld.com/article/2833267/it-management/how-box-com-allowed-a-complete-stranger-to-delete-all-my-files.html

[64] E. Bott. Dropbox gets hacked ... again. ZDNet. [Online]. Available: http://www.zdnet.com/article/dropbox-gets-hacked-again/

[65] Stilgherrian. Dropbox drops the security notification ball, again. ZDNet. [Online]. Available: http://www.zdnet.com/article/dropbox-drops-the-security-notification-ball-again/

[66] T. Espiner. One in five hacked logins match Microsoft accounts. ZDNet. [Online]. Available: http://www.zdnet.com/article/one-in-five-hacked-logins-match-microsoft-accounts/

[67] D. McMillan and D. Yadron. Dropbox blames security breach on password reuse. ZDNet. [Online]. Available: http://blogs.wsj.com/digits/2014/10/14/dropbox-blames-security-breach-on-password-reuse/

[68] Acronis. Acronis International GmbH. [Online]. Available: http://www.acronis.com/en-us/

[69] ADrive. ADrive LLC. [Online]. Available: http://www.adrive.com

[70] Bitcasa. Bitcasa. [Online]. Available: https://www.bitcasa.com

[71] Carbonite. Carbonite, Inc. [Online]. Available: http://www.carbonite.com

[72] Copy. Barracuda Networks, Inc. [Online]. Available: https://www.copy.com/companies/

[73] CrashPlan. Code 42 Software, Inc. [Online]. Available: https://www.code42.com/crashplan/

[74] Egnyte. Egnyte, Inc. [Online]. Available: http://www.egnyte.com

[75] Hightail. Hightail Inc. [Online]. Available: https://www.hightail.com

[76] IDrive. IDrive Inc. [Online]. Available: https://www.idrive.com

[77] Justcloud. JustCloud.com. [Online]. Available: http://www.justcloud.com

[78] Livedrive. Livedrive Internet Ltd. [Online]. Available: http://www.livedrive.com

[79] Mankayia. Mankayia LLC. [Online]. Available: https://www.onlinestoragesolution.com

[80] Mozy. Mozy Corporation. [Online]. Available: http://mozy.com/#slide-8

[81] Opendrive. OpenDrive. [Online]. Available: http://www.opendrive.com

[82] SOS Online Backup. SOS Online Backup. [Online]. Available: http://www.sosonlinebackup.com

[83] SpiderOak. SpiderOak, Inc. [Online]. Available: https://spideroak.com

[84] SugarSync. SugarSync, Inc. [Online]. Available: https://www.sugarsync.com

[85] ZipCloud. ZipCloud.com. [Online]. Available: http://www.zipcloud.com

[86] B. C. News. Enterprise cloud storage struggling against consumer-focused offerings, Ovum claims. [Online]. Available: http://www.businesscloudnews.com/2014/09/04/enterprise-cloud-storage-struggling-against-consumer-focused-offerings-ovum-claims/

[87] Box. Box Inc. [Online]. Available: https://www.box.com/personal/

[88] Dropbox. Dropbox, Inc. [Online]. Available: https://www.dropbox.com

[89] Google Drive. Google Inc. [Online]. Available: https://drive.google.com

[90] iCloud. Apple Inc. [Online]. Available: https://www.icloud.com

[91] Microsoft OneDrive. Microsoft. [Online]. Available: https://ondrive.live.com

[92] "Security and compliance - Office 365," White Paper, Microsoft, May 2014. [Online]. Available: go.microsoft.com/fwlink/p/?LinkId=401240

[93] Seafile. Seafile Ltd. [Online]. Available: http://seafile.com/en/home/

[94] ownCloud.org. ownCloud.org. [Online]. Available: https://owncloud.org

[95] assistant. [Online]. Available: http://git-annex.branchable.com/assistant/

[96] SparkleShare. [Online]. Available: http://sparkleshare.org

[97] Syncthing. [Online]. Available: http://syncthing.net

[98] Stacksync. cloudspaces. [Online]. Available: http://stacksync.org

[99] OpenStack. OpenStack Foundation. [Online]. Available: http://www.openstack.org

[100] K. B. Leong, "How to fit BYOD into an enterprise mobility strategy," *Network-World Asia*, vol. 10, no. 3, pp. 12–14, Sep. 2013.

[101] L. L. Peterson and B. S. Davie, *Computer networks: a systems approach*, 3rd ed. Burlington, MA: Morgan Kaufmann, 2012.

[102] J. Lau, M. Townsley, and I. Goyret, "Layer Two Tunneling Protocol - version 3 (L2TPv3)," RFC 3931, March 2005. [Online]. Available: http://tools.ietf.org/html/rfc3931

[103] S. Kent and K. Seo, "Security architecture for the Internet Protocol," RFC 4301, December 2005. [Online]. Available: http://tools.ietf.org/html/rfc4301

[104] C. Kaufman, P. Hoffman, V. Nir, P. Eronen, and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)," RFC 7296, October 2014. [Online]. Available: http://tools.ietf.org/html/rfc7296

[105] D. McGrew and P. Hoffman, "Cryptographic algorithm implementation requirements and usage guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)," RFC 7321, August 2014. [Online]. Available: http://tools.ietf.org/html/rfc7321

[106] B. Patel, B. Aboba, W. Dixon, G. Zorn, and S. Booth, "Securing L2TP using IPsec," RFC 3193, November 2001. [Online]. Available: http://tools.ietf.org/html/rfc3193

[107] K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little, and G. Zorn, "Point-to-Point Tunneling Protocol (PPTP)," RFC 2637, July 1999. [Online]. Available: http://tools.ietf.org/html/rfc2637

[108] W. Simpson, "The Point-to-Point Protocol (PPP)," RFC 1661, July 1994. [Online]. Available: http://tools.ietf.org/html/rfc1661

[109] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina, "Generic Routing Encapsulation (GRE)," RFC 2784, March 2000. [Online]. Available: http://tools.ietf.org/html/rfc2784

[110] G. Pall and G. Zorn, "Generic Routing Encapsulation (GRE)," RFC 3078, March 2001. [Online]. Available: http://tools.ietf.org/html/rfc3078

[111] B. Schneier, P. Zatko, and D. Wagner, "Cryptanalysis of Microsoft's PPTP authentication extensions (MS-CHAPv2)," in *Secure Networking-CQRE [Secure]'99*. Dusseldorf, Germany: Springer Berlin Heidelberg, Nov. 1999, pp. 192–203.

[112] "Introduction to Cisco IPsec technology," White Paper, Cisco, Jun. 2007. [Online]. Available: http://www.cisco.com/c/en/us/td/docs/net_mgmt/vpn_solutions_center/2-0/ip_security/provisioning/guide/IPsecPG1.pdf

[113] "Cisco AnyConnect secure mobility client for mobile platforms," Datasheet, Cisco, Dec. 2014. [Online]. Available: http://www.cisco.com/c/en/us/products/collateral/security/anyconnect-secure-mobility-client/data_sheet_c78-527494.pdf

[114] "Junos Pulse secure access server," Datasheet, Juniper, Nov. 2013. [Online]. Available: http://www.juniper.net/assets/us/en/local/pdf/datasheets/1000357-en.pdf

[115] J.-E. Ekberg, K. Kostiainen, and N. Asokan, "Trusted execution environments on mobile devices," in *Proc. ACM SIGSAC Conf. Computer & communications security*, Berlin, Germany, Nov. 2013.

[116] "TPM MOBILE with trusted execution environment for comprehensive mobile device security," Whitepaper, Trusted Computing Group, Jun. 2012. [Online]. Available: http://www.trustedcomputinggroup.org/files/static_page_files/5999C3C1-1A4B-B294-D0BC20183757815E/TPM%20MOBILE%20with%20Trusted%20Execution%20Environment%20for%20Comprehensive%20Mobile%20Device%20Security.pdf

[117] "TPM main specification level 2 version 1.2, revision 116 - design principles," Trusted Computing Group, March 2011. [Online]. Available: http://www.trustedcomputinggroup.org/files/static_page_files/72C26AB5-1A4B-B294-D002BC0B8C062FF6/TPM%20Main-Part%201%20Design%20Principles_v1.2_rev116_01032011.pdf

[118] Z. Zhao and F. C. Colon Osono, "TrustDroid: Preventing the use of smartphones for information leaking in corporate networks through the used of static analysis taint tracking," in *Malicious and Unwanted Software (MALWARE), 7th Int. Conf.*, Fajardo, Puerto Rico, Oct. 2012.

[119] G. Russello, M. Conti, B. Crispo, and E. Fernandes, "MOSES: supporting operation modes on smartphones," in *Proc. 17th ACM symposium Access Control Models and Technologies*, Newark, NJ, Jun. 2012.

[120] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones," in *Proc. 9th USENIX Conf. Operating Systems Design and Implementation*, Vancouver, BC, Canada, Oct. 2010.

[121] S. Chung, S. Chung, T. Escrig, Y. Bai, and B. Endicott-Popovsky, "2TAC: Distributed access control architecture for "Bring Your Own Device" security," in *BioMedical Computing (BioMedCom), ASE/IEEE Int. Conf.*, Washington, D.C., Dec. 2012.

[122] D. Titze, P. Stephanow, and J. Schutte, "A configurable and extensible security service architecture for smartphones," in *Advanced Information Networking and Applications Workshops (WAINA), 27th Int. Conf.*, Barcelona, Spain, Mar. 2013.

[123] G. Portokalidis, P. Homburg, K. Anagnostakis, and H. Bos, "Paranoid Android: versatile protection for smartphones," in *Proc. 26th Annual Computer Security Applications Conf.*, Austin, TX, Dec. 2010.

[124] "Cisco Bring Your Own Device," White Paper, Cisco, Apr. 2012. [Online]. Available: http://www.cisco.com/web/solutions/trends/byod_smart_solution/docs/BYOD_Whitepaper.pdf

[125] "Meet evolving enterprise mobility challenges with Samsung KNOX," White Paper, Samsung, Feb. 2014. [Online]. Available: http://www.samsung.com/global/business/mobile/platform/mobile-platform/knox/downloadFile/Samsung-KNOX_Whitepaper_web_Feb.27.2014-0.pdf

[126] KNOX Workspace. Samsung. [Online]. Available: https://www.samsungknox.com/en/products/knox-workspace/technical

[127] Samsung KNOX hypervisor. Samsung. [Online]. Available: https://www.samsungknox.com/en/blog/samsung-knox-hypervisor-powered-integrity

[128] KNOX Workspace supported MDMs. Samsung. [Online]. Available: https://www.samsungknox.com/en/products/knox-workspace/technical/knox-mdm-feature-list

[129] T. Richardson and J. Levine, "The Remote Framebuffer Protocol," RFC 6143, March 2011. [Online]. Available: https://tools.ietf.org/html/rfc6143

[130] T. Bektaş, J.-F. Cordeau, E. Erkut, and G. Laporte, "Exact algorithms for the joint object placement and request routing problem in content distribution networks," *Computers & Operations Research*, vol. 35, no. 12, pp. 3860–3884, Dec. 2008.

[131] VNC mobile solution. Real VNC. [Online]. Available: https://www.realvnc.com/services/mobile.html

[132] Oracle VM VirtualBox. Oracle. [Online]. Available: https://www.virtualbox.org

[133] AndroVM. [Online]. Available: http://androvm.org/

[134] Android-x86. [Online]. Available: http://www.android-x86.org

[135] Droid VNC server. onaips.com. [Online]. Available: http://www.onaips.com/wordpress/?page_id=60

[136] Mocha VNC. Mochasoft. [Online]. Available: http://mochasoft.com/iphone_vnc.htm

[137] VNC viewer. Real VNC. [Online]. Available: https://www.realvnc.com/products/ios/

[138] VNC client for iPhone, iPad and iPod Touch. Bamtoo. [Online]. Available: http://vnc.bamtoo.com

[139] Mocha VNC for Android. Mochasoft. [Online]. Available: http://mochasoft.dk/android_vnc.htm

[140] TinyVNC. Cru. [Online]. Available: http://www.windowsphone.com/en-us/store/app/tinyvnc/553dac98-46bf-47ae-b785-0139dd310925

[141] D. Etheridge, "Developing Android applications for ARM Cortex-A8 cores," Whitepaper, Texas Instruments, Mar. 2012. [Online]. Available: http://www.ti.com/lit/wp/spry193/spry193.pdf

[142] Genymotion. Genymobile. [Online]. Available: https://www.genymotion.com/

[143] Remote Desktop Protocol. Microsoft. [Online]. Available: https://msdn.microsoft.com/en-us/library/aa383015.aspx

[144] Gesture list for the touch and mouse pointer input modes for the remote desktop apps. Microsoft. [Online]. Available: http://blogs.msdn.com/b/rds/archive/2014/10/02/gesture-list-for-the-touch-and-mouse-pointer-input-modes-for-the-remote-desktop-apps.aspx

[145] T. Tan-Atichat and J. Pasquale, "VNC in high-latency environments and techniques for improvement," in *Global Telecommunications Conf. (GLOBECOM), IEEE*, Miami, FL, Dec. 2010.

[146] D. Ehringer, "The Dalvik virtual machine architecture," Tech. Rep., March 2010. [Online]. Available: http://davidehringer.com/software/android/The_Dalvik_Virtual_Machine.pdf

[147] Android NDK. Android. [Online]. Available: http://developer.android.com/tools/sdk/ndk/index.html

[148] M. Choi and S.-H. Lim, "x86-Android performance improvement for x86 smart mobile devices," *Concurrency and Computation: Practice and Experience*, 2014.

[149] B. Shneiderman, *Designing the User Interface - Strategies for effective human-computer interaction*, 3rd ed. Reading, MA: Addison-Wesley, 1998.

[150] GENI. Raytheon BBN Technologies. [Online]. Available: http://www.geni.net

[151] vncdotool 0.8.0. Python Software Foundation. [Online]. Available: https://pypi.python.org/pypi/vncdotool

[152] VNC Tight encoder - comparison results. Python Software Foundation. [Online]. Available: http://www.tightvnc.com/archive/compare.html