HONEYPOTLABSAC: A VIRTUAL HONEYPOT FRAMEWORK FOR ANDROID

Vladimir B. Oliveira, Zair Abdelouahab, Denivaldo Lopes, Mario H. Santos, and Valéria P. Fernandes

Department of Electrical Engineering, Federal University of Maranhão, São Luís - MA, Brazil

vladimir_pi@yahoo.com.br, zair@dee.ufma.br, dlopes@dee.ufma.br
marioh90@gmail.com, valpriscilla@gmail.com

ABSTRACT

Mobile *devices suffer daily threats of various kinds, in particular in a digital form, where users without consent receives and installs malware on their mobile devices via wireless networks, getting their information vulnerable to unauthorized persons. Aiming to learn more targeted attacks on mobile devices, this paper presents a mechanism that emulates services and protocols within a mobile device with the Android OS. Our work is inspired by the idea of a Honeypot and its use is to learn from the invader. Thus, we first propose a Framework for mobile devises for reuse purposes. Then, we use the Framework itself to propose a honeypot that emulates services such as telnet, http and SMS in the application level of the Android operating system*

KEYWORDS

Network Security, Intrusion detection, Honeypots, Android

1. INTRODUCTION

With technological advances, mobile devices such as smartphones, tablets, netbooks, etc.. have maintained their commercial value to an acceptable level for all social classes. Mobile devices have increasingly cornered the market and they are used on a large scale because of allied prices, ease of use, communication with social networks, access to the internet via 3G or wireless network connections, banking transaction, internet shopping, payments, among others. With all of these features aggregated to mobile devices they started to be targets of attacks ranging from viruses, trojan horses, worms and botnets. In personal computers, there is a huge range of tools divised in order to reduce cyber-attacks. One of them is a Honeypot which is a computational resource dedicated to be probed, attacked or compromised in an environment that enables the recording and control of these activities. [8]

The work aims to propose a mechanism that consists of a Framework to generate virtual honeypots running on the application level that emulates services on the Android operating system. Through the framework, we are able to extend with new services and protocols to be emulated.

The rest of this paper is organized as follows. Section 2 provides theoretical information about the Android operating system, mobile devices attacks and Honeypots. Section 3 describes the HoneypotLabsac, a Virtual Honeypot Framework for Android. Section 4 reviews related work and section 5 presents the tests and results. Finally, section 6 gives the concluding remarks.

DOI : 10.5121/ijcnc.2013.5413

2. THEORETICAL FUNDAMENTALS

2.1. Android Operating System: ARM Architecture

Android is a complete platform for mobile devices [9], in particular for smart handsets Smartphones. The platform consists of an operating system, middleware, applications, and user interface. Android is the first project of an open-source platform, with a license of Apache Software Foundation (ASF) [10]. The license of type ASF allows changes to be made to the source code but not necessarily shared [10]. The Android operating system was initially developed by Google and released into the market for mobile handsets in 2008 [9] [8]. Shortly thereafter, Google has created alliances with other companies in various segments of mobile phone market to continue the project. This alliance is called the Open Handset Alliance (OHA), which currently has 84 companies that encompass the entire structure that is part of the procee of mobile phones [1] [11].

The Android operating system was launched in the market of mobile handsets in 2008. It has become quite popular because it is an open source operating system, by providing an SDK platform for developers using JAVA programming language and a Market app with few restrictions to disseminate applications developed by third parties. The Android operating system has in its base the Linux kernel version 2.6, which has undergone with several changes to suit mobile devices (smartphone), and it is responsible for security and memory management. The linux OS is serving as an abstraction layer between the hardware components and software, is not considered a conventional Linux distribution. It uses a virtual machine called Dalvik fully optimized to work with mobile devices that run applications developed in JAVA programming language. When a Java application is compiled the (.class) program is converted to (.dex) (Dalvik Executable) thereafter files of (.dex) and other resources such as images are compressed into a single file extension (.apk) (Android Package File), which is nothing more than the actual application for Android already compiled. [1] [9].

2.2. Android Operating System : X86 Architecture

The Project of Android OS x86 is founded by Chih-Wei Huang (Taiwan) and Yi Sun (USA) [12]. The x86 architecture is a modified version of the original architecture of microprocessors Advanced RISC Machine (ARM), modified and adapted to Android x86. This system has as main objective to offer full support for netbooks and notebooks.

2.3. Cyber-attacks on Mobile Devices

Cyber-attacks in mobile devices are not new. The first virtual virus for cell phone was developed in 2004. This virus is called "Cabir" and its spread is via Bluetooth and concerned only the Symbian operating system [7]. Dunham, K. et al. [14] have defined some threats on mobile devices using a mobile security terminology which includes threats such as Ad / Spyware, Bluebug, BlueChop, bruteforce, Denial-of-Service (DoS), Exploit, Hacking, Malware Móvel , Snarf, Trojan, Virus and Worm. These threats are:

- Ad / Spyware - are potentially unwanted programs, which may include (End User License Agreement) EULA, End User License. This type of program performs various undesirable actions without user consent. It commonly involves the appearance of ads (pop-ups) and also studies the user's profile.

- Bluebug - exploits a vulnerability in Bluetooth to make phone calls with differentiated value (expensive calls).
- BlueChop - is a denial of service attack that disrupts a network of Piconet.
- Bruteforce - is an algorithm which consists of enumerating all possible outcomes of a solution. Each solution found verifies if it satisfies the problem.
- Denial-of-Service (DoS) - is an attack designed to disrupt and/or deny the use of a mobile device, or network service.
- Exploit - is a software that tries to take advantage of a vulnerability to perform undesirable actions.
- Hacking defaults - are techniques used to break devices or software they use passwords, and / or default settings.
- Malware Móvel - are softwares that perform malicious actions on mobile devices.
- Snarf - is the unauthorized data theft. It is a slang used to steal information from another device.
- Trojan - is malicious software that masquerades as something that is not in reality. Its role is to steal information and sends it mostly to the sender or to its creator.
- Viruses - are malicious software that intentionally multiplicate through programs or files on a mobile device in order to change the work form of mobile device following two criterias: self executing and self replication.
- Worm - are malicious software that creates a copy of itself, a clone, aims to spread. Most described cyber threats for mobile computers have come from PC (Desktop). These threats have undergone some modifications for use on mobile devices. Some may even be versions of the originals. But their goals are to obtain unauthorized information of the user or damage the unit.

2.4. Honeypots

Hoepers et al. [8] has defined a Honeypot as a computational security resource which is dedicated to be probed, attacked or compromised. Spitzner [6] states that a Honeypot is tool for security study where its main function is to collect information from the attacker, through monitoring. Thus a honeypot is an attractive tool placed in a network committed to monitor the attacker and the tools which it uses.

There are two types of Honeypots: Low and high interactivity [5] [8] [13]. Low interactivity Honeypots - simulates only part of some operating systems, network protocols, services and some commands with which the attackers may interact, thus minimizing the risk of compromising the actual operating system. High interactivity Honeypots allow the attackers interact directly with the operating systems, applications or real services. Some authors such as Spitzner [6] makes a reference to medium interactivity honeypots that lies between low and high interactivity ones. In table 1, we show a comparison between levels of Honeypot interactivities.

Security community faces daily several challenges. One of them is to obtain valuable information through the data that is collected. Enterprises collect daily large amounts of data, including firewall logs, system logs, and intrusion detection alerts. However, all this amount of data can become just another bunch of hits. On the other hand, honeypots collect a small amount of data, but this is of great value. Honeypots are not part of the system production. In fact, all traffic destined to them is considered as malicious. The traffic can be considered of high value and can be used in statistical models, periodic reviews, attack detection, or even on offensive research methodologies [6].

Level of interactivity	Facility for installing & configuring	Facility for activation & maintenance	information collection	Level of interactivity
Low	Easy	Easy	Limited	Low
Medium	Medium	Medium	Variable	Medium
High	Difficult	Difficult	Extensive	High

Table 1 – interactivity Levels, Spitzner, [6]

Some caracteristics and features of honeypots need to be defined and these are: monitoring, audit log, containment and visibility. Monitoring - A Honeypot has as its main function monitoring network activities. Audit log - is vital to maintain it in an intact file to rebuild attacks in case they occur. Containment - Honeypots are meant to be committed, but a compromised honeypot cannot serve as a bridge for further attacks. Visibility - a Honeypot is more attractive when it is visible because it can draw more attacker attention.

3. HONEYPOTLABSAC: A FRAMEWORK FOR ANDROID VIRTUAL HONEYPOT

3.1. HONEYPOTLABSAC Framework

Initially, the proposed framework is inspired by some works that have already been developed in [3] [4] [5], which aims at providing a virtual Honeypot [5] and a virtual honeypot for mobile device [3] [4]. Differently from others, our proposal aims simply on collecting data from a mobile device connected to a wireless network through a virtual Honeypot installed on the Android operating system.

The FrameworkLabsac generates a virtual Honeypot for the Android operating system, which can later be expanded to other operating systems for mobile devices. The goal of creating the Framework is directly related to software reuse, because the project can be extended at any time by adding new services and protocols to be emulated. As we can see, the FrameworkLabsac has a simple architecture composed of 03 (three) packages: br.labsac.honeypot.core, br.labsac.honeypot.util br.labsac.honeypot.

- **br.labsac.honeypot.core** - This package consists of five (5) classes: GeneralService, HoneypotProtocol, ProcessAdminRequests, ProtocolProcessing and ProtocolResponse. The class GeneralService is responsible for providing the connection via socket to customers, in other words it is the class server connection. The HoneypotProtocol class emulates a connection. The class ProcessAdminRequests is responsible for checking whether a particular service is running. The ProtocolProcessing class checks whether or not a service should be stopped. The ProtocolResponse class responds to it clients requests using its methods.
- .**br.labsac.honeypot** - This package implements three (3) classes: HoneypotActivity, HoneypotApplication and HoneypotStarter. The HoneypotActivity class is the base class

for the core program. The HoneypotApplication class is responsible for setting preferences. The HoneypotStarter class is responsible for checking network connectivity by starting the service.

- **br.labsac.honeypot.util** - this package has three (3) classes: ArchiverInServerAlarm, MsgType, and ServiceImage. The ArchiverInServerAlarm class is responsible for generating the file in sdcard of the mobile device to record the logs and then send the generated file to the server logs. The MsgType class is responsible for all types of generated messages. The ServiceImage class is responsible for the serialization of communication with the ports and their methods.

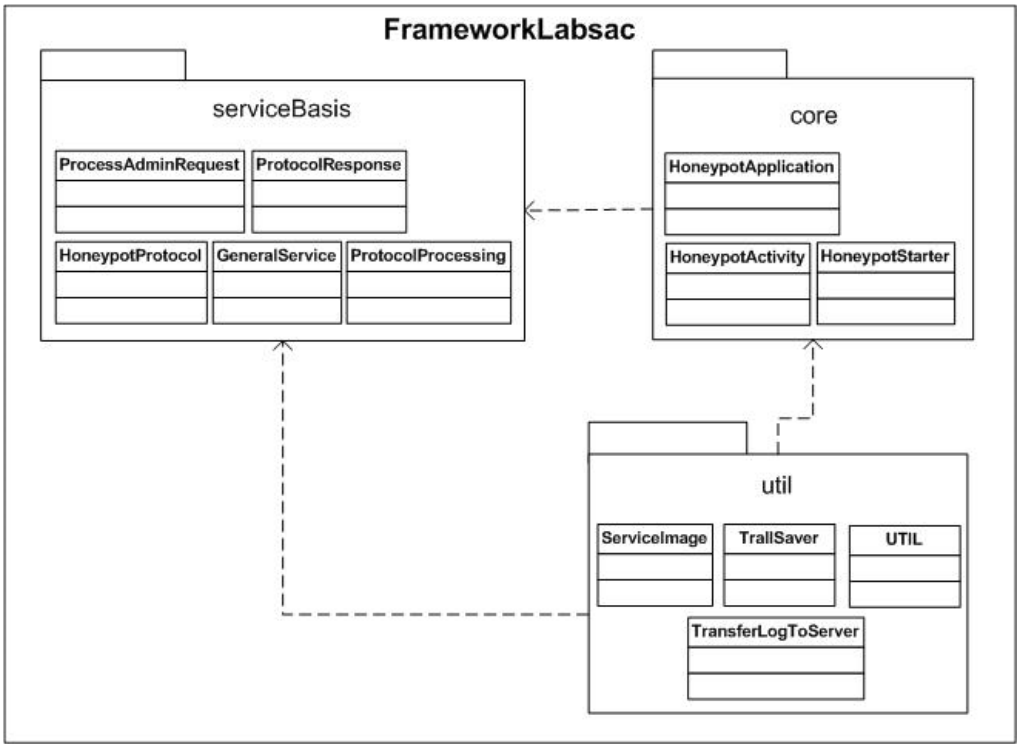


Figure 1: Package Diagram of FrameworkLabsac

The following section explains how to extend our framework with new protocols or services.

3.2. HONEYPOTLABSAC Application

The HoneypotLabsac application is designed to run on the Android operating system at the application level and to emulate telnet, SMS, and http services. As a result, a log file of all interactions and accesses is generated.

Aiming to solve the difficulties encountered in relation to Honeypot, we have implemented the application with the following measures:

- **Monitoring**: the application intends to monitor the entire events in order to recreate an attack in case it occurs.
- **Log for audit**: to maintain the integrity of the log file, this one is sent from time to Containment: when log file is sent to the log server and the latter detects that a virtual

machine is compromised, an autonomic service is activated to create new firewall rules and a new virtual machine to replace the machine.

- **Visibility**: Mobile devices are increasingly connected to wireless networks. HoneypotLabsac is designed to emulate popular services such as telnet, SMS, and http aiming to draw attention of the attacker.

In Figure 3 we present the architecture of HoneypotLabsac.

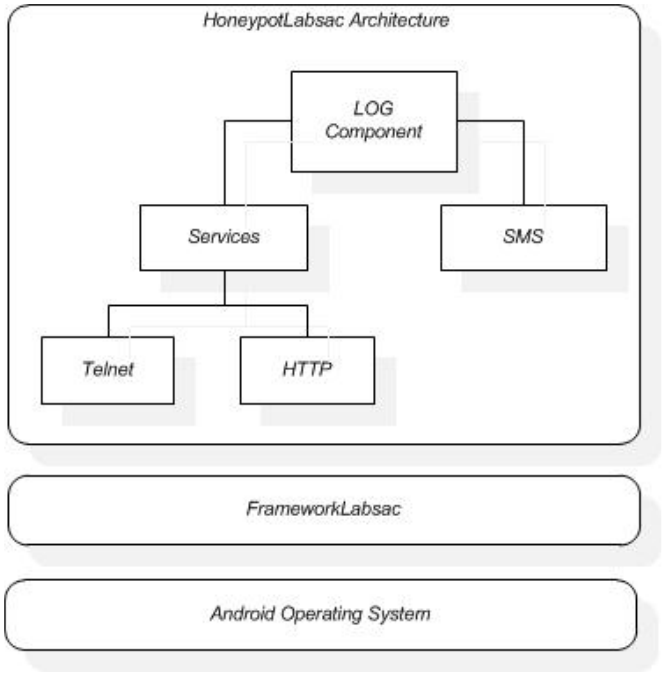


Figure 2: Architecture Diagram of HoneypotLabsac

We can observe from Figure 2 that the HoneypotLabsac architecture consists of few components: LOGComponent, Services, and SMS.

- **LOGComponen**t - this component is responsible for generating and storing all information generated through the connections of Telnet and HTTP as well as the SMS received from the operating company of cell phones. Thus, a log file for each service emulated by HoneypotLabsac.
- **Telnet Service** - aims to emulate the Telnet service of the Android operating system. By emulating the service, it gives a feeling to the attacker that he is accessing the real Telnet service of the Android operating system.
- **HTTP Service** - aims to emulate the HTTP service, by returning fake information to the attacker.

The HoneypotLabsac is implemented using Java technology [44], the integrated development environment IDE (Integrated Development Environment) Eclipse [43] and the Android SDK [2] [21].
Figure 3 shows the class diagram of HoneypotLabsac and its relationships.

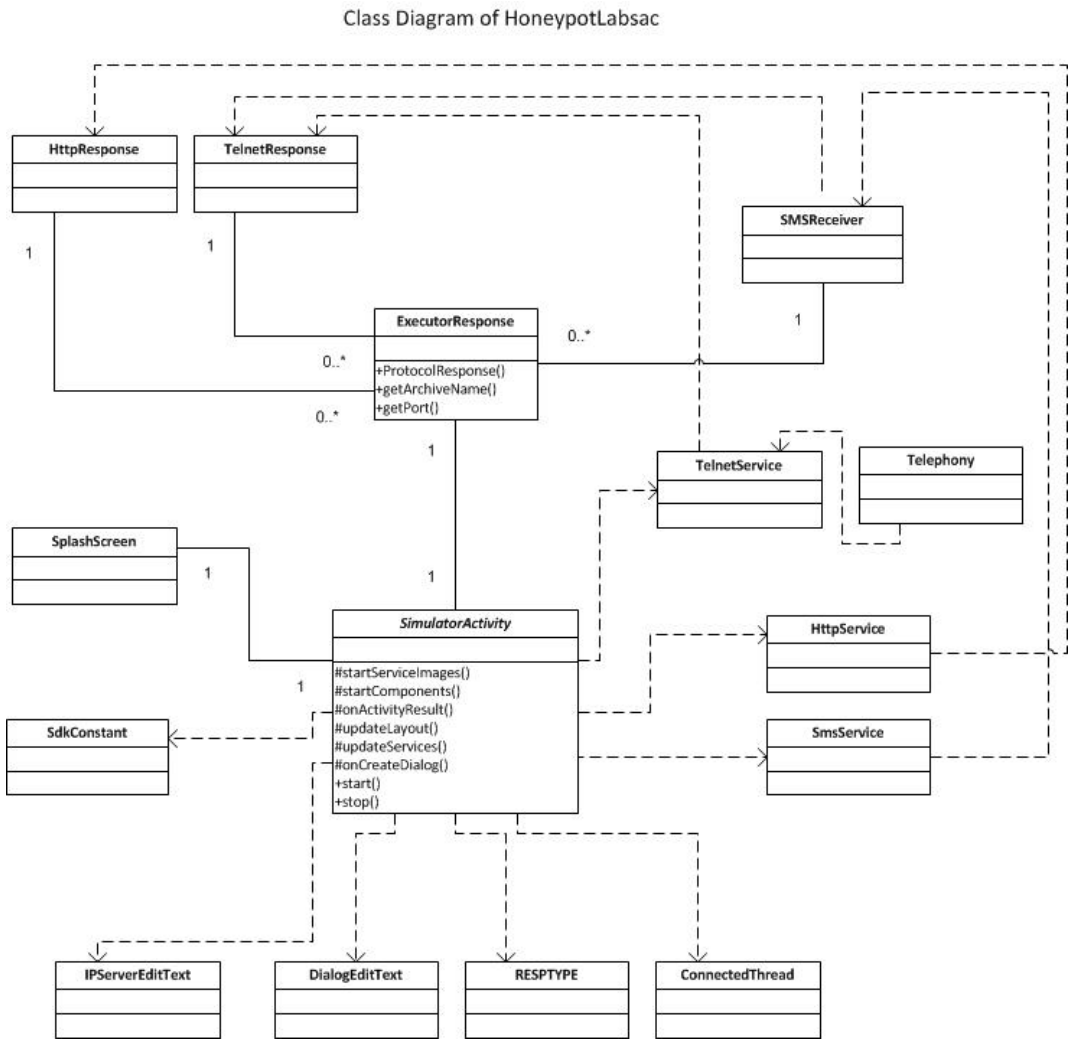


Figure 3: Class Diagram of HoneypotLabsac

Figure 4 shows a use case for capturing data with HoneypotLabsac. When HoneypotLabsac is active (running in background), any interaction with Telnet, HTTP, sending or receiving SMS through Telnet connection or through the cell phone company log files are generated. These log files are stored primarily in the internal memory of the mobile device and after they are sent to the IDS NIDIA log server.

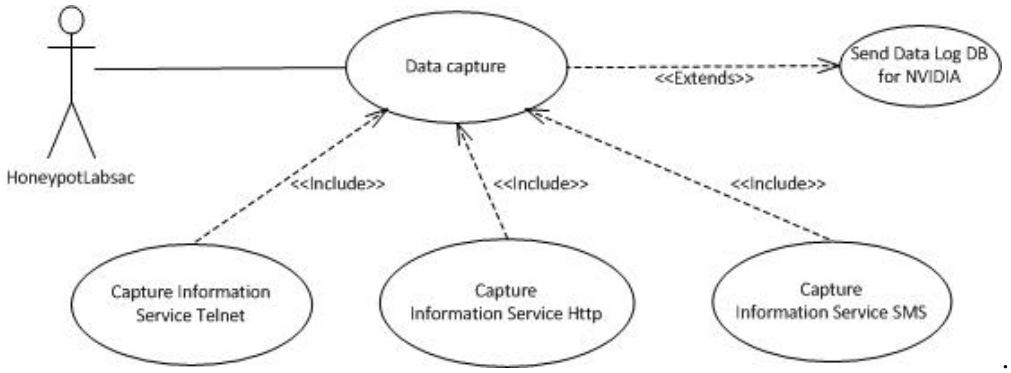


Figure 4: Use Case Diagram for Traffic Capture

3.3. Using HONEYPOTLABSAC

The HoneypotLabsac is a specific tool for the Android operating system and it is easy to install and configure. The settings are made through the application screens, where it is possible to choose which service to emulate and which communication ports these services should use. It is also possible to configure the IP number of the log server and the communication port between the log server and the mobile device. The HoneypotLabsac is a tool for network administrators who want to discover any attacker via a mobile device connected to their networks.

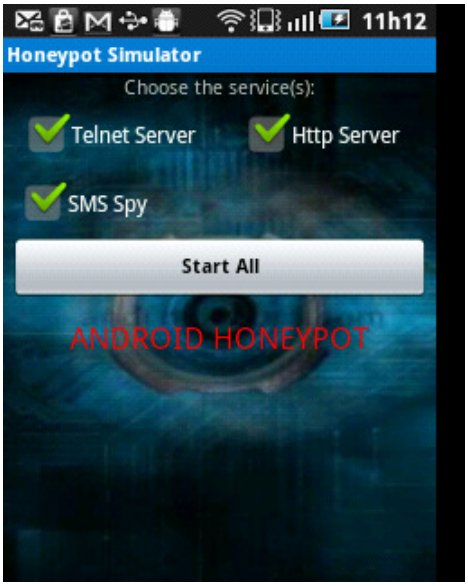


Figure 5: Main Screen of HoneypotLabsac

Figure 5 shows the main screen of HoneypotLabsac, presenting the available services that can be emulated and the corresponding checkbox to mark them. Figure 6 shows a screen with active services, by selecting the "Start Marked" ', after such services are marked in the main screen.

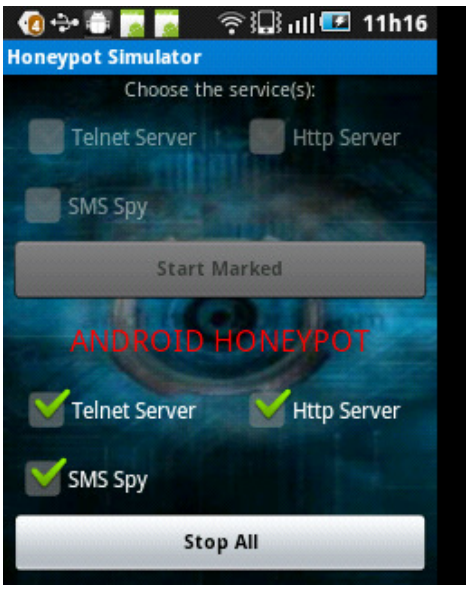


Figure 6: Screen Settings of Services Initialization in HoneypotLabsac

Figure 7 illustrates a screen with settings made for the communication ports for HTTP and Telnet services, as well as the configuration of the IP number and port number of the log server. The transfer of log file from the mobile device to the log server occurs only after this setting.

Figure 7: Configuration Screen of Communication Ports of HTTP and Telnet services

4. RELATED WORKS

Provos [5] has developed a framework called "Honeyd" which emulates multiple virtual honeypots and multiple signatures of computer systems at the network level. These signatures are known as Fingerprint. Each Honeypot can emulate the signature of an operating system on a real

machine. The Honeyd supports IP protocol, and responds to requests from the network of virtual honeypots according to the services that were configured on each one.
Collin (et al) [3] creates a Honeypot for smartphone with Android operating system called Honeydroid at the real kernel device level. It uses virtualization of the hardware such as modem, wifi and sdcard, to generate records of attacks. According to the author this is the first Honeypot for a real mobile device.

Oconnor (et al 2010) [4] have developed "honeyM", a framework for virtual Honeyclient, emulating mobile Bluetooth, infrared, GPS, 3G, and WiFi. It is developed in python and is used to emulate an Apple iPhone.

Barrera [2] analyzes popular platforms of mobile devices from 2010 such as: Android, BlackBerry, iPhone and Symbian, and presents a classification of installing third party software on smartphones, using three generic models: Walled Garden Model, Guardian Model and End-user Control Model.

Characteristics	Honeydroid	HoneyM	Honeyd	HoneypotLabsac
Framework	No	Yes	Yes	Yes
Application Level	No	No	No	Yes
Operating System Level	No	No	Yes	No
Kernel Level	Yes	No	No	No
Hardware Vitualizacion	Yes	No	No	No
Low Interactivite	No	Yes	Yes	Yes
High Interactivite	Yes	No	No	No
Android Operating System	Yes	No	No	Yes
Others Operating System	No	Yes	Yes	No
Mobile Operating System	Yes	Yes	No	Yes

Table 2: Comparison

To compare the works directly related to HoneypotLabsac, we have taken into account the following characteristics:

- Framework – aiming software reuse.
- Application Level - applications that are installed in mobile operating systems.
- Level Operating System - emulate operating systems signatures known as Fingerprints.
- Kernel Level - the layer below the operating system.
- Hardware Virtualization - Works with virtualized hardware.
- Low Interactivity - Provides services and signatures of operating systems --- false (Fingerprints).
- High Interactivity - Provides real operating systems, applications, services.
- Android Operating System - Works directly with the operating system Android.
- Mobile Operating System - Works with mobile operating system.

From the comparison ilustrated in Table 2, we can observe that HoneypotLabsac is the first Honeypot for the Android operating system at the application level.

5. TESTES AND RESULTS

Figure 8 shows the scenario where the HoneypotLabsac is installed on a real Samsung Smartphone with the following configuration:

- Brand - Samsung;
- Model - Galaxy 5;
- Model number - GT-I5500B;
- Firmware Version - 2.2.2;
- Kernel Version - 2.6.32.9-perf-se.infra set-46 # 1;
- Build number - FROYO.VJJP9;
- Processor - 600Mhz.

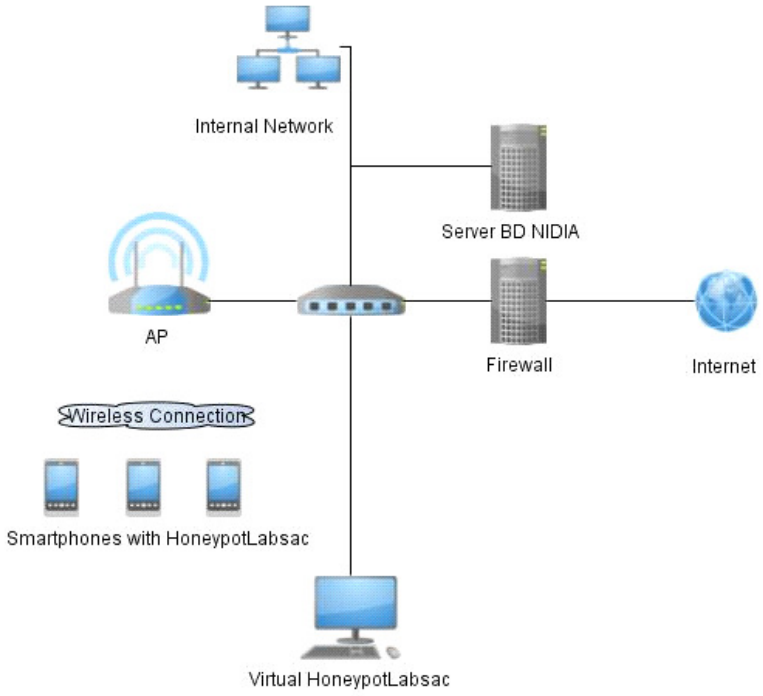


Figure 8 Test Environment

The tests done using the scenario of Figure 8 are performed in a controlled environment. This scenario is typical to someone who uses a Smartphone. This type of user is connected to the Internet through a wireless network as it becomes financially cheaper than 3G connection since no additional cost is incurred. These wireless networks are mostly at the work place, home of residence or living environments. The HoneypotLabsac is installed in the device on all its available services (Telnet, HTTP, SMS) are activated. Thus, the application is ready to intercept interactions and thus generating log files. Figures 9, 10, 11 show log files generated for each emulated service as well as the received SMS.

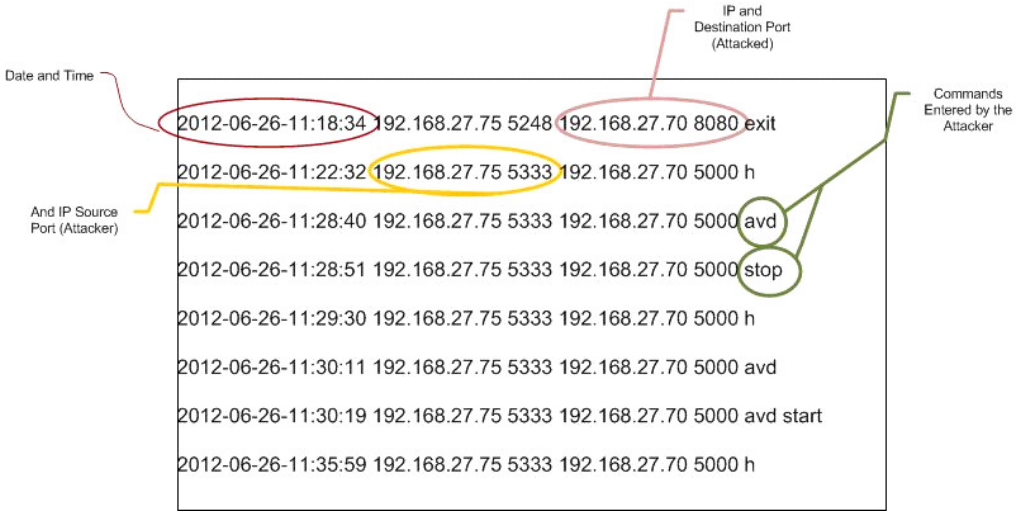


Figure 9 Telnet Log

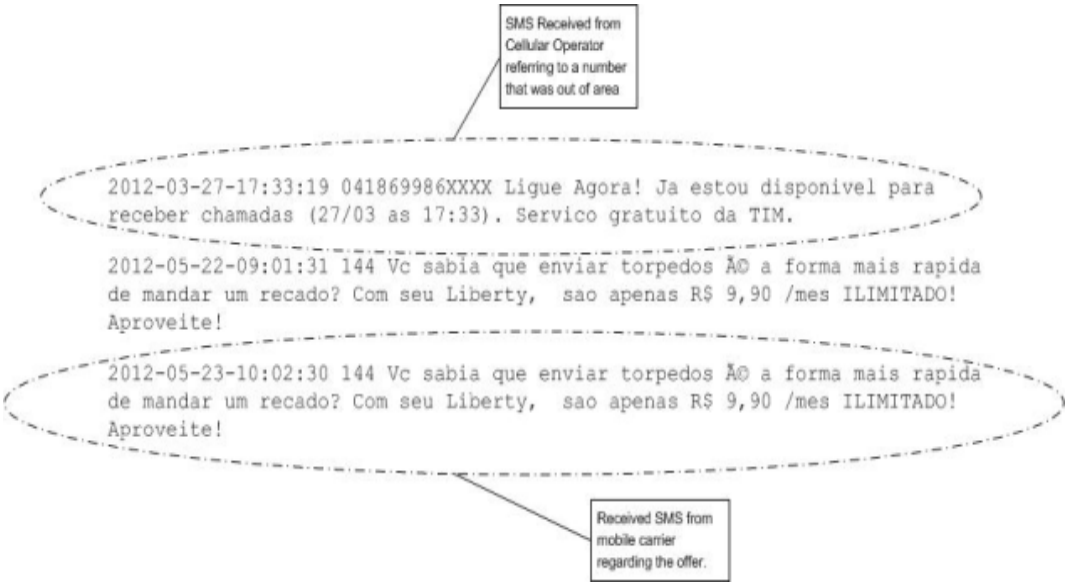


Figure 10 SMS Log

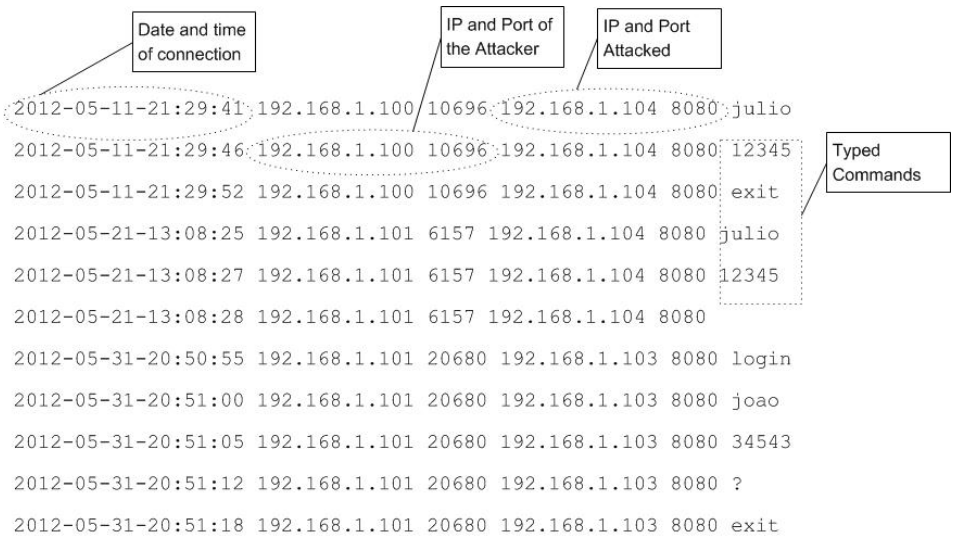


Figure 11: Http Log

6. CONCLUSION

With HoneypotLabsac, we have the opportunity to study the attacks that are made to mobile devices through wireless network and thus take some action in response to attacks. By adding new services or protocols to be emulated by the framework, the Honeypot becomes more attractive to the attackers.

From this initial work, we have identified some possibilities for future work that could be developed, such as:

- In this work the focus is the Android operating system, nothing prevents to adapt the project to run on other mobile operating systems.
- New services may be added and emulated on mobile devices.

ACKNOWLEDGEMENTS

Financial supports of FAPEMA and CNPq are gratefully acknowledged.

REFERENCES

[1] ANDROID. (2012) Sistema Operacional Android. Available in: <http://www.android.com>. Accessed on: 01/03/2012.

[2] BARRERA, D. & OORSCHOT, V. P. (2011) “Secure Software Installation On Smartphones”. Security Privacy, IEEE, v. 9, n. 3, p. 42 –48.

[3] COLLIN MULLINER, S. L.& LANGE, M. Poster, (2011) Honeydroid - Creating A Smartphone Honeypot. In: IEEE Symposium on Privacy and Security, Poster. Oakland California: IEEE,.

[4] O’CONNOR, T. J.& SANGSTER, B., (2010) Honeym: A Framework For Implementing Virtual Honeyclients For Mobile Devices. In: WiSec ’10: Proceedings of the third ACM Conference Onwireless Network Security. New York, NY, USA: ACM, pp 129–138.

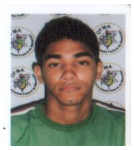
[5] PROVOS, N.& HOLZ, T., (2010) Virtual Honeypots: From Botnet Tracking To Intrusion Detection. Addison-Wesley Professional.
[6] SPITZNER, L.(2002) Honeypots: Tracking Hackers. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.
[7] SYMBIAN. Sistema Operacional Symbian. Available in: <http://licensing.symbian.org/>. Accessed on: 01 mar. 2012.
[8] HOEPERS C.; JESSEN, K. S. &. C. M. H. P. C. Honeypots E Honeynets: Definições E Aplicações. Available in: <http://www.cert.br>. Accessed on: 01/05/2012.
[9] PEREIRA L. C. O & SILVA, M. L.,(2009) Android Para Desenvolvedores. Rio de Janeiro: Brasport, 2009.
[10] LECHETA, R. , (2010) Google Android: Aprenda A Criar Aplicações Para Dispositivos Móveis Com O Android SDK. Sâo Paul o: Novatec.
[11] MEIER, R., (2010) Professional Android 2 Application Development. 1st. ed. Birmingham, UK: Wrox Press Ltd.
[12] X86, A. Sistema Operacional Android Arquitetura x86. Available in: <www.android-x86.org>. Accessed: 02/08/2012.
[13] ASSUNCAO, M. F. A. (2009) . Honeypots e Honeynets: Aprenda A Detectar E Enganar Os Invasores. Rio de Janeiro: Visual Books.
[14] DUNHAM, K. (2009) Mobile Malware Attacks and Defence,. Burlington: Elsevier,.
[15] ECLIPSE. Eclipse. Available at: <http://www.eclipse.org.>. Accessed on: 01/06/2012.
[16] DEITEL, H. M.& DEITEL, P. J. (2003) Java Como Programar. 4. ed. Porto Alegre: Bookman,.

Short Biography

Vladimir Olivwira has obtained her B.Sc. degree in Computer Science from UESPI, Brazil in 2000 and his M.Sc. degree in Computer Science from UFMA, Brazil in 2012. His research interests include networking, and security.

Zair Abdelouahab is a professor in Computer Science at the Federal University of Maranhão (UFMA) in Brazil. He is conducting research on networks and security. He received his BSc degree in Computer Engineering from University of Setif, Algeria in 1985. He has an M.Sc. degree in Computer Science from Glasgow University, UK in 1988 and a Ph.D. degree in Computer Science from Leeds University, UK in 1993.

Denivaldo Lopes is a lecturer at the Federal University of Maranhão (UFMA) in Brazil. He received his B.Sc. degree in Electrical Engineering from UFMA in 1999. He obtained his M.Sc. in Electrical Engineering and Computer Science, from UFMA in 2001 and his Ph.D. degree in Computer Science from the University of Nantes/France in 2005. His research interests include Service-Oriented Architecture, Model Driven Engineering, and Security**.**



Mario Braga is currently finishing a BSc degree in Computer Science at the Federal University of Maranhão (UFMA).

Valeria Fernandes is currently finishing her BSc degree in Computer Science at the Federal University of Maranhão (UFMA).