Building a Hybrid Experimental Platform for Mobile Botnet Research

Apostolos Malatras, Laurent Beslay
European Commission, Joint Research Centre (JRC), Institute for the Protection and Security of the Citizen
Email: apostolos.malatras@jrc.ec.europa.eu, laurent.beslay@jrc.ec.europa.eu

This paper was presented at Botconf 2015, Paris, 2-4 December 2015, www.botconf.eu
It is published in the Journal on Cybercrime & Digital Investigations by CECyF, https://journal.cecyf.fr/ojs

DOI: 10.18464/cybin.v1i1.8

***Abstract*—Mobile botnets are an emerging security threat that aims at exploiting the wide penetration of mobile devices and systems and their vulnerabilities in the same spirit of traditional botnets. Mobile botmasters take advantage of infected mobile devices and issue command and control operations on them to extract personal information, cause denial of service or gain financially. To date, research on countering such attacks or studying their effects has been conducted in a sporadic manner that hinders the repetition of experiments and thus limits their validity. We present here our work on a hybrid experimental platform for mobile botnets that supports the execution and monitoring of related scenarios concerning their infection, attack vectors, propagation, etc. The platform is based on principles of flexibility, extensibility and facilitates the setup of scalable experiments utilising both real and emulated mobile systems. We also discuss a novel method of estimating the active bot population in a botnet and illustrate its deployment on the experimental platform.**

I. INTRODUCTION

The ever increasing capabilities of modern smartphones and the advanced functionalities that they offer to their users, e.g. by making use of the plethora of embedded sensors, have spurred the growth of mobile applications and have widened the avenues of computing to have a more pervasive nature. Moreover, the integration of the mobile ecosystem with the traditional computing one has created many opportunities, while at the same time it has introduced a series of challenges and increased the attack vectors available to malicious entities. Accordingly, security threats that are widespread in traditional networked computing systems have found their way in the mobile environment. In this respect, mobile botnets have emerged as a serious security threat that is being exploited by malicious attackers to extract lucrative gains and cause service disruption [1]. Since mobile phones are almost ubiquitously connected and tightly linked to users' personal information, there is a significant motivation for malicious entities to attack such devices and platforms [2].

Mobile botnets have spurred from traditional ones and therefore bear several similarities with them, albeit having certain differences attributed to the features of the undelying platforms and Operating Systems (OSs). Mobile botnets take advantage of the security vulnerabilities that exist in mobile OSs and their component-based architecture that builds on extensible systems with the use of apps that exacerbate security concerns [3]. Mobile botnets are coordinated and managed by the botmasters, who define the target, motivation and functionality of the botnet [4]. Their goal is to install an application or a service, i.e. a malware, on a victim's mobile phone in order to be able to "command and control" (C&C) it remotely. The botnet malware reports back to the botmaster with personal or financial information, e.g. photos or banking account details, whereas one cannot neglect possible keylogging attacks or attacks aiming at Denial of Service (DoS) [5].

Research on mobile botnets and in particular on how to detect them and take them down presents numerous challenges. It is hindered by the fact that such botnets are highly distributed, installed on devices with ad hoc connectivity and dynamic IP addressing, utilizing a series of side channels for communications, e.g. NFC or Bluetooth, as well as malware masquerading as legitimate applications or services. A major difficulty consists in not being able to effectively replicate realistic mobile botnet infection and distribution in the lab and thus conduct proper experimentation. To date there have been few efforts in this direction, which would provide significant input on the behaviour of mobile botnets and assist in countering their adverse effects. In this paper, we propose such an experimental research platform that can be used for tests and trials of mobile botnet research efforts, thus promoting applicable solutions and addressing the needs of the research community.

We present the design and implementation of a hybrid mobile botnet research platform for experiments. The proposed platform is hybrid in that it makes use of both actual mobile devices, as well as emulated ones. This design requirement was necessitated by the fact that we aim to provide a solution that would enable scalable and flexible experiments, while at the same time allowing to monitor the realistic behaviour of botnets on mobile devices. Furthermore, we examine how the platform can be used in conducting experiments by describing an experimental approach to measure the size of a botnet. We make use of the Jolly-Seber capture-recapture method [6] widely adopted on the field of biology, and adapt it to the particularities of mobile botnets. Our ongoing experimental analysis plans to validate both the effectiveness of the platform, as well as that of the size estimation approach and its performance.

The remaining of this paper is structured as follows. After this brief introduction, Section II describes mobile botnets, their components and functional features. Section III discusses the goal of the experimental platform, its design requirements and Section IV its implementation, whereas its limitations are also underlined. In Section V we present a use case of an experiment that can be conducted on the platform. The experiment is based on a novel approach to count the number of active bots in a mobile botnet. Finally, Section VI reviews related work in the area of mobile botnets experiments and size estimation, whereas the paper concludes with Section VII where the limitations of this work are underlined and opportunities for further research in the domain are pinpointed.

II. MOBILE BOTNETS

A botnet is a collection of compromised machines that aims to perform certain activities based on the desires of its creator, namely the botmaster [4]. Botnets are generally used for malicious purposes by the botmaster in order to disrupt the operation of services or extract lucrative gains from unsuspected users. The communication between the botmaster and the compromised machines usually takes place over the Internet or another type of network, hence the definition of botnets as networks of bots [7].

The high-level architecture of botnets is broadly comprised of the following fundamental components:

- **Botmaster:** The owner/initiator of the botnet, who is in charge of defining the functionality of the botnet, its action and is the recipient of any relevant lucrative operations. The botmaster is usually defining the high-level nature of the attack (the functionality of the botnet), whereas this is translated to actual commands by the C&C Server (commands differ according to the targeted platform). Recruiting campaigns for new bots are among the objectives of the botmaster [8].
- **C&C Server/infrastructure:** It is responsible for sending commands for controlling and contacting the bots to retrieve information. It has knowledge of the bots that are under its control and can communicate with them. The connections to the bots need not always be open, but can be activated when new commands are issued to the bots. The C&C Server executes the functionality defined by the botmaster in the botnet. The C&C infrastructure can be owned by the botmaster or abused by him/her, e.g. IRC server or online social network used to distribute commands to the bots [9]. Depending on the architecture of the botnet, there might be more than one C&C Server for a mobile botnet.
- **Bots:** End-user devices that have been infected by the botnet malware and are thus susceptible to receive commands and controls from the botmaster, via the C&C Server. We can distinguish between:
  - **Servant bots:** Servants forward commands received by the C&C or other servants to client bots. The standard bot functionality, i.e. execution of received commands and reporting back to the C&C, also applies to servants.
  - **Client bots:** Client bots are the last link in the chain of control of a botnet. They are listening for commands from the C&C Server and they report back to the C&C Server or their delegated servant.
- **Communication channels:** They refer to the networking infrastructure available to the devices involved in a botnet. Accordingly, these can be exploited for a series of different functionalities, such as the initial infection of the botnet, its propagation, the commands issued by the botmaster and the C&C server and the information collected and reported by the bots. Different communication channels impose their own constraints in regards to the various botnet architectures [10][11].

There are various aspects in regard to the functionality and operation of botnets that can be used to classify them and such aspects have undoubtedly a different essence when examining traditional botnets in comparison to the more recent mobile ones. In our previous work [2], we built a taxonomy of botnet features and examined how they apply to both mobile and traditional botnets. The features that we considered in our taxonomy are:

1) **Network/Connectivity:** 3G cellular connections follow a strict hierarchical design, where there is few flexibility. The 3G network standards impose architectural constraints (centralized architecture) and the network infrastructure is owned by telecom operators and therefore it is out of the control of the end users, including the botmaster and the owners of the compromised bots. Cellular networks are not free of charge, therefore users might be able to detect strange behaviour in their accounts, e.g. number of SMS messages or exceeding data plan, which could be attributed to malicious botnet behaviour [12]. Other types of mobile botnets operate on short-range communication networks, e.g. Bluetooth or NFC. This implies temporary close proximity of the different components of botnets, and therefore they do not refer to wide scale infection scenarios. However, such networks are outside of the telco providers control since they are formed on an ad hoc basis, thus making them extremely difficult to detect and counter.
2) **Platform:** The most common mobile botnet platforms include the typical mobile OS, i.e. Android, iOS, Windows Mobile, Symbian, etc. Since Android is today the most widespread OS it is naturally the target of choice, and does indeed receive a lot of attention from botnet developers: depending on observers, 80 to 95% of malware discovered on malware platforms are Android based [3]. There still exist some fundamental differences: it is always possible for an Android user to install software that does not come from official market places, whereas it is not possible on iOS platforms, which definitely adds to the risk potential.
3) **Architecture:** Mobile botnets have particularities in that the number of exchanged messages/data between botmaster and slaves should be reduced, since it can be

subject to charges and affect the battery that could be noticeable by users. Moreover, the fact that 3G networks in particular are tightly controlled by telecom operators could deter the deployment of P2P-like architectures on their networks. Such constraints do not favor the deployment of complex architectures, such as hierarchical ones, especially when SMS is the medium of propagation. In addition, P2P architectures can only be realistically achieved when IP communication is enabled, since using SMS in multi-hop communications is not considered to be viable due to the great number of exchanged messages [13].

4) **Propagation of Infection:** Mobile devices, with the plethora of on-board sensors and the collection of a large number of personal data over time, constitute an ideal environment for context- or user-driven propagation of the infection. Consistent with the architectures that are most applicable for mobile botnets, flooding for propagation of botnets is not a realistic option when SMS is the medium of propagation, with one-to-many being a preferred choice. Selective flooding could be supported by the use of Bluetooth or NFC (or the display of QR codes) to initiate infections in some specific locations or in an advanced cases by means of on-board sensors [14].
5) **Infection Means:** Means of botnet infection for mobile devices are expected to be the same as the ones for commodity computers. Mobile devices have additional channels of communications that are unique to them and have been already exploited by botmasters, namely SMS and Bluetooth. Being aware of the decreasing rate of SMS exchanges, consideration will be given to the possibility to exploit SMS-alternatives, e.g. instant messaging, to infect mobile devices.
6) **Motivation/Impact:** Evidently, the same types of motivation that can be found in traditional botnets can also be considered for mobile botnets. Additionally, the ubiquitous nature of mobile devices that nowadays hold a huge amount of personal information, i.e. financial, photos, contact lists, etc., has made them extremely attractive for malicious attacks that enable gaining access to such data. Moreover, the fact that premium SMS services and calls can be made by smartphones, allows the attackers to gain financially from mobile botnets, an aspect that is not explicitly (or not anymore because of the disappearance of landline modems) available to commodity PCs [12]. Phones are also often attached to a market place account that could be authorized to make purchases with a preconfigured credit card number. Mobile platforms are often the tool used to receive or generate one-time passwords.
7) **Detection:** In addition to the standard techniques used for traditional botnets, in mobile botnets, since applications are installed on smartphones and granted permissions to access local resources, the notion of application analysis should be extended to cover aspects such as whether the granted permissions are actually used [15]. Moreover, the notion of honeypots that is a key detection mechanism for botnets should be reconsidered to take into account the particularities of mobile environments in terms of architecture, network, application markets, etc. [16].
8) **Target:** Conversely, the inherent particularities of mobile phones as carriers of physical environment sensors, enable more targeted selection of potential slaves. For example, devices under a specific geographic region could be targeted, or device owners with specific patterns of movements or even people that have the same set of electronic devices at home (magnetometer could be used for relevant identification). This grouping of device owners and consideration of physical world concepts could therefore be utilized to target selectively the distribution of the botnet, as well as the C&C communications, e.g. by issuing different commands to different sets of slaves.

Evidently, the list is not meant to be exhaustive. Instead, it aims at highlighting the main elements of botnets that allow to fully describe it and understand its operational behaviour. For a thorough analysis of the taxonomy we refer the interested reader to [2].

III. HYBRID EXPERIMENTAL PLATFORM

We describe in what follows the design goals of the proposed hybrid experimental platform for mobile botnets research and present its architecture. The section concludes with an enumeration of the functional features of the platform.

A. *Design Goals*

The hybrid experimental platform for mobile botnet research needs to be generic in that it should allow for a great variety of experiments to be conducted, under different settings and with diverse objectives. In this respect, flexibility of the platform is a major design goal. The platform should allow for experiments considering diverse populations of bots, namely with varying size. The setup of the experiments should cater for this diversity by supporting the definition of both simple and complex scenarios involving the infection, distribution, operation and detection of mobile botnets. In addition, the platform should be easily extensible so as to be able to support prospective enhancements and modifications without significantly affecting its operation and design. A rigid architectural design would make it extremely cumbersome to modify and extend the platform. Since the platform needs to cater for a variety of experiments, all configuration settings should be modifiable dynamically and at runtime in order to gain from maximum flexibility.

Scalability support is another important design goal. The number of bots in a mobile botnet is usually rather high and thus realistic research scenarios should include a corresponding number of devices. The platform is comprised of both actual and emulated devices, but while there is an upper limit to the actual devices included (due to costs) there is no such limit for the emulated ones. Size of the botnet will thus be only limited by the available resources, i.e. actual mobile devices in the infrastructure and memory constraints regarding the parallel execution of multiple emulators. Since mobile devices' emulators are typically consuming a large number of computational resources, the servers on which these emulators operate might become overloaded. Accordingly, load balancing should be taken into account to avoid adverse situations. Scalability should also apply to the number of distinct botnets that can be tested concurrently on the platform. While attention should be given to avoid starving resources, the platform should be implemented in a manner that does not hinder the parallel execution of more than one botnet.

In terms of the expected functional behaviour of the platform, it should support both complex and simple experiments that users can define in a straightforward manner. With the well-established diversity and heterogeneity of mobile phone OS, platforms and device capabilities, the experimental platform should also cater for this heterogeneity. Heterogeneity should also be considered in the context of networking, allowing for diverse communication protocols and network standards to be considered. Another design goal of the platform is its ability to monitor and keep track of the execution of the experiments so as to be able to produce results for a posteriori analysis.

B. Architecture

To satisfy the aforementioned design goals we opted for a modular architecture that inherently supports extensibility and flexibility. According to the particular requirements set on the platform, certain modules can be activated or deactivated to achieve the desired functionality. The high-level functional architecture of the hybrid experimental platform can be seen in Figure 1.

A fundamental requirement is to have an experimental platform on which botnet research works can be applied and tested in a realistic manner. Therefore, support for heterogeneity is important, namely being able to test the different applications on a variety of mobile platforms. Since it is very difficult to keep track of all possible mobile OS configurations, we opted for a hybrid solution where there coexist actual mobile phone platforms with emulated ones. The latter are based on the Android Emulator and we plan to extend our platform in the future by integrating emulators and actual mobile phones from different platforms. Emulation provides the opportunity to deploy a series of diverse mobile phone configurations without incurring the corresponding acquisition costs and making the experiment more manageable.

The main element of the architecture is the *Experiment Manager*. This module allows users to setup and execute their experiments on the platform and collect the related results. Users can select where their experiments will be executed, i.e. on emulated or real nodes, and setup the desired network topology according to their needs. The *Experiment Manager* displays a listing of supported mobile OSs and devices, as well as their capabilities so that users can select the number of nodes they wish to experiment with and set them up in their desired topology.

The *Experiment Manager* has 4 optional submodules that can be activated on demand according to the type of experiment. Complex experimental settings might require specific mobile phone configurations for different parts of the experiments, e.g. platform, memory, CPU, etc. and moreover might necessitate the dynamic assignment of tasks to mobile phones, as well as their migration. For this reason, we have introduced the *Task Analyzer* and *Task Allocation* submodules that are respectively responsible for breaking down the experiment into tasks and allocating them to the most suitable entity of the platform, be it emulated or real device. Examples of tasks might refer to installing the botnet malware in some devices, modifying node connectivity to reflect user movement or the assignment of C&C duties to specific devices. The *Optimization* submodule keeps track of the execution of the experiment and automatically makes appropriate adjustments to maintain the desired level of operation. Such adjustments might require moving tasks from one device to another, providing more resources to emulated devices or offloading task activities to other devices that have idle resources. Lastly, the *Objective Evaluation* submodule provides functionality to ensure that quantifiable objectives that have been described in the experimental settings are accurately fulfilled, e.g. maximum CPU utilization due to botnet malware, minimum number of infected devices in a botnet, etc.

At the core of the actual experiment execution lies the *Emulation Manager* module. It receives codified instructions from the *Experiment Manager* on how to run the experiment on the actual and emulated mobile platforms. The *Emulation Manager* communicates with these platforms and can instruct them to perform certain actions having root privileges on them. In parallel, it can receive feedback from these platforms, which is mainly used to collect results regarding the experiment as well as to monitor its proper execution and performance metrics. Accordingly, the *Emulation Manager* makes use of 4 optional submodules to complete its functionality. The *Setup* submodule is in charge of translating the codified instructions received from the *Experiment Manager* into platform-specific commands. The significance of this submodule lies in the fact that the experimental infrastructure comprises various heterogeneous platforms and each one of them has a distinct way of listening to and executing commands. In this respect, the *Execute* submodule sends the commands to the actual or emulated platforms.

Moreover, the *Settings* module maintains a record of the current experimental settings, e.g. frequency of data collection or listing of IP addressing and routing table, and can be used to modify these changes if required. A particularity of mobile botnets involves the use of sensor data to contextualize the attack, e.g. to target only devices in a specific location. Whereas the actual phones of our platform have embedded sensors that can be used to produce relevant data, the emulated ones lack such functionality. In this respect, the *Settings* module is also responsible for generating realistic sensor data

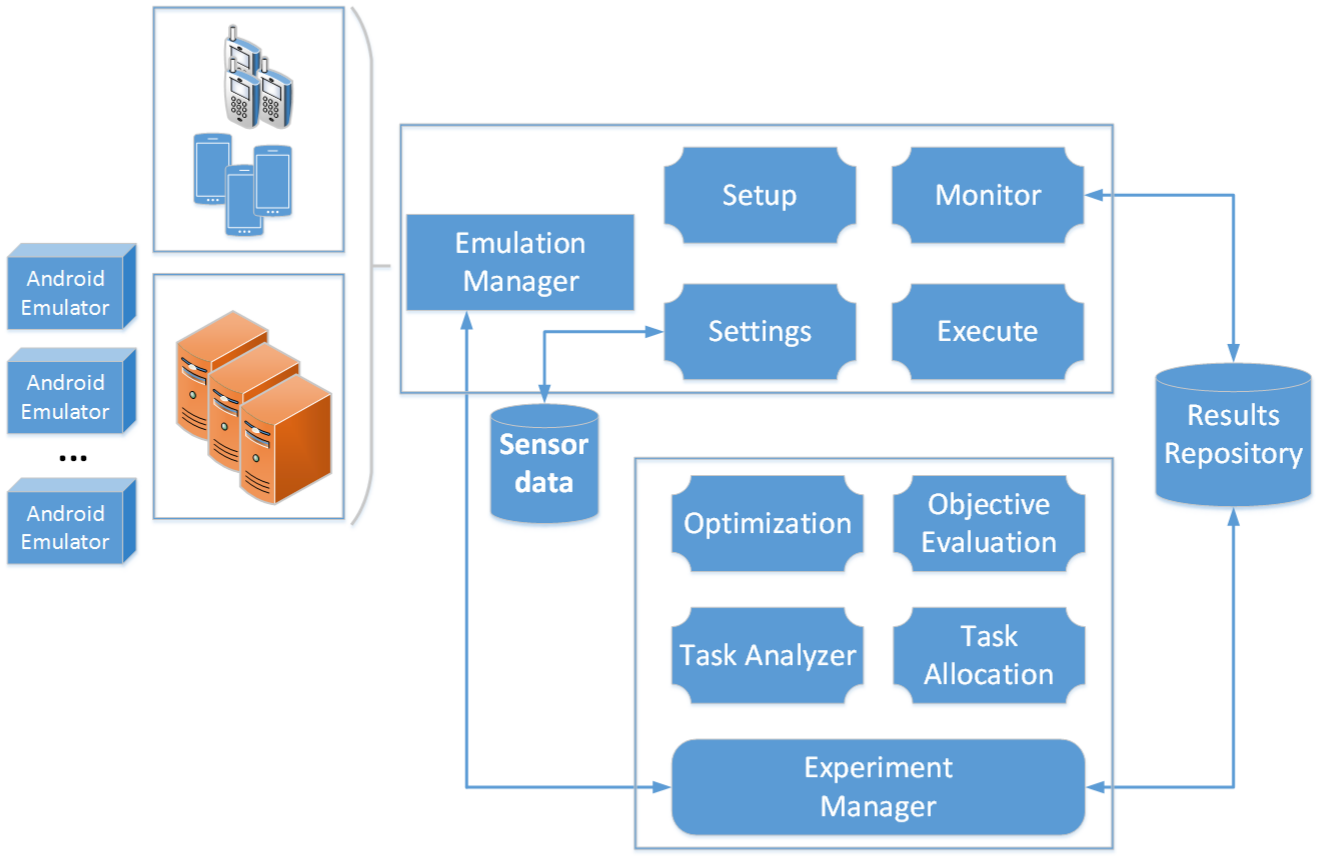

Fig. 1. Architectural design of the hybrid experimental platform for mobile botnets.

(stored in the *Sensor Data* repository) for each of the emulated Android platforms in accordance to the specifications defined in the description of the experiment. The Android Emulator allows for sensor data to be injected during the emulation in order to trigger corresponding actions, and we plan to exploit this functionality. Lastly, the *Monitor* submodule listens for the feedback from the experimental platform's devices, as well as monitors the performance and operation of the platform in line with the objectives that have been defined in the *Objective Evaluation* submodule.

The results collected from the experiment (via the *Monitor* submodule and the *Emulation Manager*) are stored in the *Results Repository* where they can be accessed by the *Experiment Manager* to be presented to the users. Furthermore, the *Results Repository* can be utilized independently by the users to analyze both the results of experiments, but also the operation of the hybrid experimental platform itself, subject to appropriate metrics having been defined and monitored.

C. Functional features

The hybrid experimental platform to support experiments regarding mobile botnets supports the following functional features:

- Hybrid emulation platform to test mobile botnets' distribution, infection, detection, etc. with configurable parameters such as the size of the botnet and the types of mobile systems.
- It supports the observation of the operation of mobile botnets in hybrid configurations (emulation and actual mobile platforms).
- It supports the setting up of experiments regarding mobile botnets using a scenario-based stepwise approach. Experiments can be as simple as launching a mobile botnet on a set of devices and monitor its evolution, to more complex configurations such as ones involving network partitions, contextualization of mobile botnets infection using sensor data, etc.
- It supports remote configuration of the emulated and real devices[1] to enable dynamic features such as topology reconfiguration, enabling/disabling of features, modification of operations, etc.
- It supports collection of results and measurements regarding mobile botnets' operation, e.g. infection rate, CPU and memory utilization, number of exchanged messages.
- It supports the integration of realistic sensor data (for the emulated devices) in order to experiment with particular mobile botnets' settings.
- It supports the parallel execution of multiple experiments

[1]To support remote configuration of real devices they should be running under the development mode and be part of the same network as the hybrid platform. In some cases, for more advance remote configuations to be possible a telnet daemon needs to be running on the real device.

(and therefore multiple mobile botnets) subject to the availability of the desired resources, i.e. emulated and actual mobile systems.

IV. IMPLEMENTATION

Based on the aforementioned design requirements and architecture specification, we implemented a prototype of the hybrid experimental platform. In the following, we discuss practical aspects in regard to the platform's implementation, as well as its limitations and future enhancements.

Implementation of the platform is based on three pillars:

1) **Infrastructure:** the platform is based on a series of hardware components that enable the emulation of mobile botnet experiments. The platform comprises both actual mobile devices to test botnet malware functionalities, as well as multiple Android emulators that are operating on servers in order to cater for scalability issues. Interconnecting real and virtual Android devices is a challenging issue, especially when modifications need to be applied in order to exhibit features of dynamicity.
2) **Software environment:** development of the platform needs to take into account the design requirements set in the previous section. In accordance to the need for flexibility, extensibility and modularity, the object-oriented programming paradigm was selected as the most suitable option and in this respect the functional components of the platform were implemented using Java to take advantage of its inherent features.
3) **Configuration:** the platform needs to accommodate a variety of experimental configurations. We thus opted for a configuration framework based on XML settings files that are used to define both the setup of the platform, as well as the experiment's settings. The platform monitors XML settings files at runtime and therefore modifying the latter files would lead to a dynamic adaptation of the experimental platform itself.

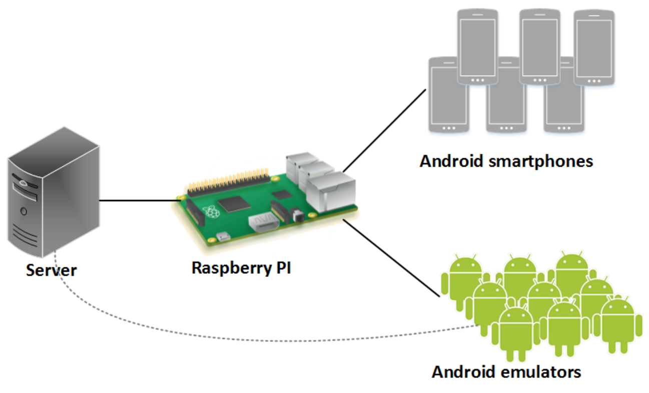

Fig. 2. Elements of the infrastructure for the prototype's implementation.

A. Infrastructure

The prototype of the hybrid experimental platform comprises a rich set of infrastructure components, displayed in Figure 2, in order to emulate the largely diverse mobile botnet ecosystem. In particular, the prototype consists of the following elements:

- **Server:** It is the system used to execute the prototype and control the hybrid experimental platform, while at the same time it is where the Android emulators are instantiated and running for the purposes of various experiments. In the current deployment cycle we have provisioned a sole server, but we plan to expand to multiple ones in the future. The server is running on a machine with two Intel Xeon E5-2670 octacore processors at 2.60 GHz and 32GB of RAM, allowing for the parallel execution of numerous emulators (in our experiments we were able to have up until 20 concurrent instances of the Android emulator operating).
- **Mobile devices:** 6 Android smartphones were used in the prototype to test its functionality, namely 2 Samsung Galaxy Nexus, 2 Sony Xperia LT26i and 2 HTC One X, all running Android version 4.2.1. While the phones were equipped with active SIM cards, for the initial round of experiments we utilized their WiFi connections for connectivity to facilitate integration with the rest of the platform and work in a more contained environment. The use of 3G connection is a straightforward transition for the platform, since communications in any case take place over IP.
- **Raspberry PI:** A Raspberry PI 2 Model B is used to perform the functionalities of the C&C Server. The motivation behind the use of this equipment was to allow for a machine with adequate resoirces (900MHz quad-core ARM Cortex-A7 CPU and 1GB of RAM) to play the role of C&C Server, while at the same time enriching the diversity of the mobile botnet ecosystem. Moreover, we wished to physically separate the C&C Server from the server used to setup the experiment, the latter being conceptually conceived as the machine used by the botmasters to launch their attack.
- **Networking:** The experimental platform requires networking connectivity between all participating elements to exit. While actual mobile botnets in the wild utilize cellular connections, as well as wide Internet ones, we opted for a constrained testing environment to be able to control it better. For this reason we have set up a WiFi network in our lab, access to which is available to the server, the Raspberry PI and the mobile devices, whereas we have restricted its Internet access. Emulated devices acquire IP addresses through the virtual router that is included within each instance of the Android emulator. To enable connectivity between emulated instances and real mobile devices we use standard network port redirection to fix the egress/ingress links between them.

All these components of the hybrid experimental platform are part of the same WiFi network, in order to contain the ecosystem and isolate it from the outside world. More importantly, in order to be able to use the Android *adb* utility and thus have connectivity and interaction with the actual devices, it is essential that all infrastucture elements are in the same WiFi network. One way to support devices being in different networks for example would be to have a telnet daemon running on all mobile devices of the platform in order to enable remote connectivity. Moreover, it is evident that the platform can be easily expanded to include further devices of any kind, subject to memory constraints and costs, as well as more C&C Servers.

B. *Software environment*

Tha hybrid experimental platform has been implemented using open-source technologies and in particular using Java to benefit from its inherent support for flexible, extensible and modular applications. For the main functionality of the platform there is no need to utilize any packages external to the standard Java distribution. All types of settings and data are stored in XML databases and these can be handled by default and support for advanced networking is also possible using the Java networking facilities. Lastly, interaction with the actual operating systems, for example to instruct a mobile device to change its routing table or to install a specific software such as a mobile botnet malware, is supported through the Java Native Interface (JNI).

To optimize the performance of the platform we have made use of various software patterns. The singleton pattern was used to restrict the instantiation of classes that refer to objects that according to our design requirements should have only one active instance. This refers to the *Experiment Manager* and its submodules, as well as the *Emulation Manager*, but as far as the latter is concerned not all of its components are instantiated once. The *Execute* and *Monitor* submodules have multiple instances, one for each emulated and real device that partake in the experiment. Access to the *Results Repository* is facilitated by means of a dedicated component, namely the *Results Manager*, to which the singleton pattern is also applied.

One of the particularities of mobile botnets is the fact that they can exploit sensor information to contextualize the attack and even to personalize it to specific user sets. The experimental platform that we have implemented cannot address user personalization issues in regard to the devices, real or emulated ones, since it is extremely cumbersome to create and maintain - by means of mere programming - user profiles with realistic information. This is a drawback of the current implementation and we are exploring ways on how to alleviate it. In terms of the sensor data, we have provided capabilities to integrate such data using the *Settings* submodule of the *Emulation Manager*. Sensor data on emulated devices are provided by utilizing the open-source utility SensorSimulator[2]. It allows the simulation of sensor data, namely accelerometer, compass, orientation, temperature, light, proximity, pressure, gravity, linear acceleration, rotation vector and gyroscope sensors, according to desired settings such as the frequency of data generation.

[2]https://code.google.com/p/openintents/wiki/SensorSimulator

Moreover, location being one of the most important information in terms of contextualizing mobile botnets attacks, special consideration has been given to the simulation of realistic location information on both the actual and the emulated phones. In regard to the latter, the aforementioned utility is used to provide location information to the emulated devices. The changes in the location of an emulated device are time-dependent and can be triggered on demand by the *Emulation Manager*. Conversely, in real devices it is once again the responsibility of the *Emulation Manager* according to the configuration settings of the experiment. On all the mobile devices that are part of the experimental platform's ecosystem we have enabled the "Allow mock locations" permission on Android and have created an app (implementing a mock location provider) that modifies the current location when a certain period of time has elapsed, i.e. 1 minute. The locations are prefined and stored in an XML file. With these mechanisms in place we are able to emulate the mobility of real and emulated devices and thus support mobile botnet experiments with advanced location-aware functionalities.

C. *Configuration*

To define and run experiments users need to create a configuration file in XML in accordance with an XML Schema that we have devised. Experiments are defined using a scenario-based approach, where a sequence of steps is specified to reflect the different stages of the experiment. The steps can be sequential in order and triggered at particular point in time, whereas there is also the provision to have conditional triggering for steps in line with monitored conditions or results of the experiment itself, e.g. when a certain percentage of the node population has been infected with the mobile botnet malware or when a certain amount of data has been collected respectively. The user can at any time pause the execution of the scenario or modify it by adding or removing steps, namely parts of the experiment. We opted for a SAX (Simple API for XML) instead of a DOM (Document Object Model) parser in the prototype's implementation. Accordingly, the XML configuration file is not fully loaded upon initialization, but instead loads one scenario step at a time, thus facilitating dynamic modification of the experiment.

Let us assume an experiment where we plan to explore the operation of a mobile botnet and to monitor the exchange of messages between infected devices up until a certain number of messages has been observed. In the configuration file we first need to define some identifier information for the experiment. The two foremost elements of the configuration consist of the node list and scenario steps. The former is the list of nodes -from those available to the platform- that will be used for the experiment. When listing the utilized nodes the user has to also define their neighbors, thus constructing the desired network topology. The scenario steps describe the sequence of events that constitute the experiment. In this case, the first step would be to infect all participating devices, real and

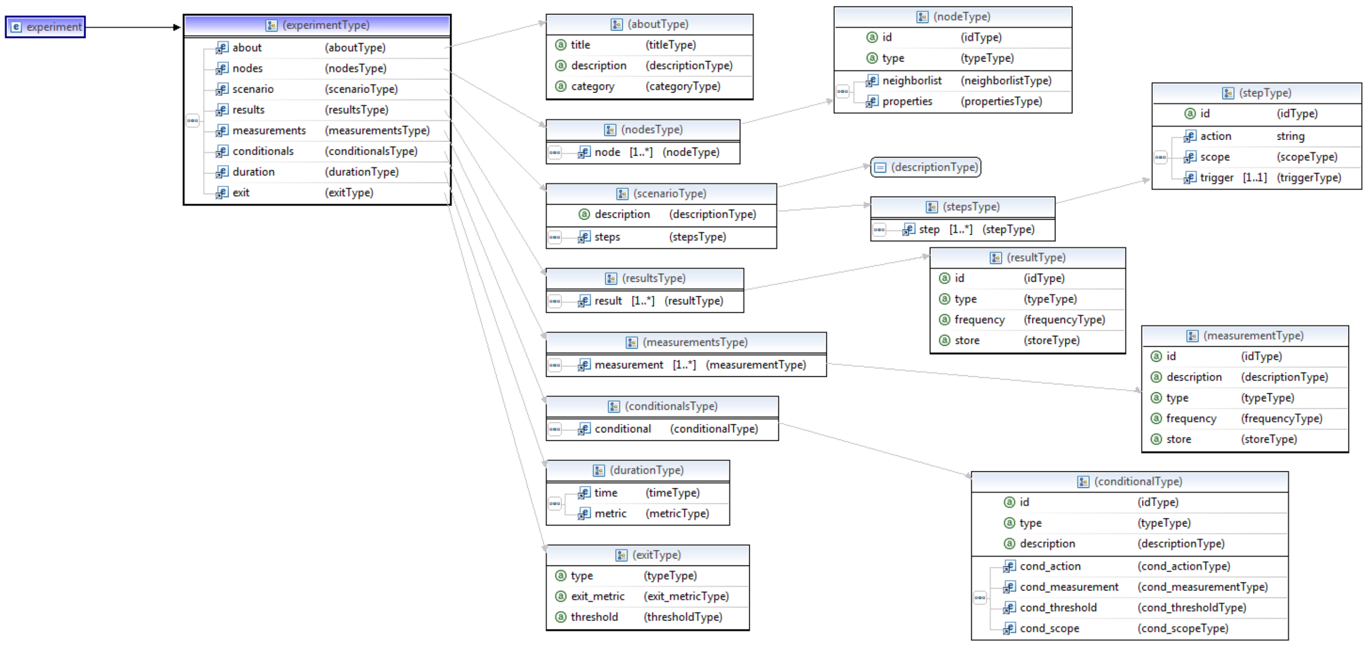

Fig. 3. XML Schema for the configuration of mobile botnets experiments.

emulated, with the mobile botnet malware, namely install an APK file. We would then need to manage the interconnections between devices (using port redirection techniques) to form the desired topology and then to setup monitors for the exchanged messages. Other elements of the experimental configuration include the collection of results, i.e. what types of data to monitor and record, as well as the exit condition, which can be either in the form of a predefined duration or expressed by means of a conditional. In the example presented before the conditional is linked to a specific result, namely that of exchanged messages.

Figure 3 illustrates the XML Schema used for the configuration of experiments on the hybrid platform. It is interesting to note that the notion of actions, i.e. commands and instructions to be applied on the real and emulated devices, has both a triggering and a scope element. These respectively refer to the condition upon the occurence of which an action will become effective and the nodes on which it will be applied. There are two types of entries in regards to the monitoring of the experiment's progress and operation. Results are final measurements that are permanently stored for post-experiment analysis, whereas measurements are typically exploited to define conditional statements in regard to the activation of scenario events.

V. MOBILE BOTNET EXPERIMENTS

In this Section we discuss how to use the proposed hybrid experimental platform to conduct experiments regarding mobile botnets. To test the validity of the platform we plan to test and evaluate an algorithm that addresses one fundamental issue of mobile botnet research, namely that of estimating the number of the active bots.

A. *Background*

The mobile botnet population estimation algorithm is built on the principles of the Jolly-Seber capture-recapture method [6] that has been widely used in the field of biology to calculate the size of animal populations. The algorithm is based on capturing a set of items of the population, marking them and then releasing them. In a second round, the process is repeated and the population can be estimated assuming that the items are evenly distributed in the population. Evidently, the more capture-marking rounds, the better the estimation. In the context of mobile botnets, we plan to emulate a mobile botnet with many active nodes. Our solution aims at P2P mobile botnets, where each node is aware of a certain subset of the overall nodes in the botnet. Therefore, we plan to exploit certain nodes in the botnet and have them infiltrate the botnet and become on purpose infected. These nodes who will be under our control, will continuously collect information on the botnet's active bots and we can thus approach the problem in a similar manner as the Jolly-Seber method.

While there have been efforts on estimating the size of botnets in the past, such works have either not focused on mobile environments or were based on probabilistic models [17], thus limiting their validity in realistic contexts. Of particular interest is the work by Rossow et al [18] that discusses the modeling of P2P botnets by studying and analyzing several botnet families. They propose two methods of estimating the size of a botnet, namely using crawling techniques to traverse a botnet's nodes and the infiltration of the botnet by so-called sensor nodes that passively discover infected nodes and their neighbors. In this respect, the capture-recapture approach also exploits the notion of sensor nodes to divulge the internal workings of a botnet, as well as using the neighbor lists to crawl the botnet, albeit being stochastic in nature in order to reach an estimate in a short amount of time.

The capture-recapture method has proven to be quite efficient in similar contexts, e.g. calculating the number of malicious domains in fast-flux service networks [19], and has been also applied in estimating the size of P2P botnets [19]. Nonetheless, the latter research works utilise a simplistic version of the capture-recapture method by Jolly-Seber that does not take into account deaths and new entries in the population. Considering the dynamic nature of botnet infections, as well as that of the mobile environment, it becomes evident that as far as mobile botnets are concerned the simple version of the Jolly-Seber method would prove to be inefficient. We therefore propose to apply the complete method proposed in [6] and consider new infections that add to the mobile botnet population, as well as the departure of infected nodes that can be attributed to the crashing of devices (DoS) or network disconnections.

B. Stochastic model

The stochastic model to estimate the population of mobile botnets is based on the capture-recapture theory (CRT) that was introduced in [6]. For this reason, we consider the nodes that comprise the model as one species that forms an open population, i.e. nodes can join the botnet at any time (new infections) or exit it (devices crashing or botnet infection purged). Our goal is to be able to calculate the following metrics:

- N_i, total number of nodes in the population when the ith, $1 \leq i \leq t$ sample has been considered.
- ϕ_i, probability of an infected node remaining part of the botnet during sampling period i, with $1 \leq i \leq t$.
- B_i, number of new nodes joining the botnet duting the sampling period between the i and $i+1$th sample.

According to the CRT, to calculate these metrics we need to conduct a sequential sampling where at each step we mark individual observed instances of botnet infections out of a bigger sample. Assuming sampling is random and that the distibution of infected devices over the entire set of devices is uniform then with a second such sampling we can infer the total number of infected devices in the population. This is accomplished by using the ratio a_i of infected devices over the total ($a_i = m_i/n_i$, m_i is the number of marked samples at step i over a total of n_i samples) and the total number of marked animals M_i which can be calculated using the following equations:

$$M_i = \frac{s_i * Z_i}{R_i} + m_i$$

$$Z_i = \sum_{j=1}^{t} a_{j,i-1}$$

$$a_{ij} = \sum_{j=1}^{t} n_{ij}$$

$$R_i = \sum_{j=i+1}^{t} n_{ji}$$

n_{ij} represents the number of nodes in sample i that were last captured in sample j, while s_i refers to the number of nodes that were marked and released at sampling period i.

The three metrics that we wish to estimate can therefore be calculated as suggested in [6] as follows:

$$N_i = \frac{M_i}{a_i} \tag{1}$$

$$\phi_i = \frac{M_{i+1}}{M_i - m_i + s_i} \tag{2}$$

$$B_i = N_{i+1} - \phi_i(N_i - n_i + s_i) \tag{3}$$

Using equations (1)-(3) we can have a progressive estimate of the number of nodes in a mobile botnet, assuming there exists a mechanism to monitor it and collect data about nodes, namely a mechanism to simulate the capture-recapture procedure.

C. Application on mobile botnets

Capture-recapture methods work extremely well when dealing with various species populations, which was their initial application domain. Porting these methods in the mobile botnet realm requires some additional considerations in order to make them both applicable and usable. How to capture and mark instances of devices infected with mobile botnets and keep track of the tagging process is the key to succesfully realize the latter goal. In [19], the authors consider such a method oriented towards P2P botnets, where certain devices infiltrate a P2P mobile botnet and record the number of other nodes they connect to. The entries corresponding to each device reflect a marking period, i.e. nodes that are commonly seen by two devices are marked as recaptured ones. While such a method yields interesting results, it nonetheless has limited scope since the marking of infected devices is instantaneous and does not consider dynamics in the node list maintained by an infected device. Moreover, this static snapshot of a mobile botnet might not be sufficient to map the botnet's topology, since the devices used to infiltrate the botnet are located in the same subnet, namely that of the research testbed.

Conversely, we propose to exploit the inherent characteristics of P2P botnets where the node list of infected devices is regurarly updated. In addition, we are using devices infected with mobile botnet malware that are periodically reset and assigned IP addresses from different subnets therefore ensuring a more rich dataset regarding the botnet's active population. In terms of capturing and marking samples, the policy adopted involves monitoring the node list on each node and periodically extracting snapshots that serve as the capture and marking phases. We thus aim to avoid the aforementioned pitfalls and shortcomings.

To implement the proposed capture-recapture method for the estimation of botnet poupulation in our hybrid experimental framework we proceeded in a stepwise manner (thisn would be reflected on an appropriately defined XML configuration file), namely:

- We utilize only real mobile devices to infect them with the mobile botnet malware (emulated ones might be blocked from participating to the botnet).
- The desired topology is setup by means of port redirection techniques, where node neighborhood information is utilized.
- The mobile devices are infected with the mobile botnet malware, i.e. it is installed on them by the *Experiment Manager*.
- The *Task Analyzer* and *Task Allocation* modules are employed to instruct the mobile devices to reset and modify their networking configurations periodically. The *Objective Evaluation* module is additionally instructed to keep track of the CPU utilization of the devices and stop the experiment if it exceeds a threshold currently set at 40%, as well as to reset networking settings of a device if and only if at least one set of results has been obtained from that device.
- The *Emulation Manager* will receive the instructions from the *Experiment Manager* and using the *Setup* submodule will install the malware on the devices by calling on the *Execure* submodule. The *Settings* submodule is in charge of modifying networking configuration and the *Monitor* one records the other botnet nodes that the malware interacts with (in the case of P2P botnets it is the nodes in the peer list).
- Results (listing of nodes encountered in the mobile botnet) are stored in the *Results Repository*.
- At each round of new results from all devices (n_i) of the experimental platform, the stochastic model's equations are applied to calculate the estimated size of the population. When a node has been discovered it is assumed as marked (m_i), while upon second observation as recaptured and marked again.

While the process is easier applicable on mobile P2P botnets (since peer lists yield significant information regarding other nodes in the botnet), we can foresee ways to alleviate this shortcoming in the case of centralized or hierarchical/hybrid botnets, e.g. by installing specialized honeypot software on the mobile devices and then infecting them with the malware. This area of study is part of our future exploratory work.

VI. RELATED WORK

In line with currend trends in information society towards integrated mobile and commodity computing systems, botnets are expanding to new territories and moving to new modes of communication and platforms such as smart mobile devices [20], [12], [21], [13]. Mobile botnets exploit the security vulnerabilities that exist in mobile OSs and their component-based architecture that builds on extensible systems with the use of apps that exacerbate security concerns [1]. Moreover, the ever increasing capabilities of mobile networks and that of mobile platforms both attest to the convergence of commodity systems with mobile ones. Accordingly, this established paradigm shift gives ground to the recent proliferation of mobile as well as hybrid botnets [3].

Mobile botnets have certain particular features that clearly distinguish them from their traditional counterparts and necessitate novel solutions specifically targeted to address them. Contextualization is a key ingredient of the mobile ecosystem and its exploitation by botmasters opens up a series of challenges. The context of users, which is possible to infer and access thanks to the rich sensor set available to mobile phones, e.g. location, proximity, etc. can be utilized by malicious entities to target specific user groups [22]. Moreover, the sensors themselves pose a significant security threat as there has been research work on using them to bypass standard communication channels and therefore to transparently send command and control instructions to mobile phones [14].

Research work on mobile botnets has witnessed a growth in the last years, attributed to the corresponding increase in related security threats. Mobile botnet detection solutions that are targeted at the mobile ecosystem have emerged, e.g. honeypots [23], [24]. Conducting experiments in regard to mobile botnets has nonetheless not been examined in a systematic manner by the research community. Different approaches have been exploited, such as simulations, ad hoc configurations, and collection of real world statistics. We argue that in order to have systematic research on mobile botnets, experiments should be performed systematically, allowing for them to be repeated in a contained environment. Such an approach is quite common for other fields, e.g. widespread emulation platforms such as PlanetLab [25] or simulation environments such as OMNET++ [26], but has not to date been studied in the specific context of mobile botnets. Our proposed hybrid experimental platform aims at providing a generic framework for mobile botnet research works to be tested and validated in a homogeneous manner.

VII. CONCLUSION

To properly study and analyze mobile botnets, it is necessary to conduct experiments in a systematic manner. Related experiments need to take place in such a way that their repetition is possible in order to facilitate the exchange of research results among members of the corresponding community. We proposed in this paper and implemented a hybrid experimental platform targeted at mobile botnets, in order to promote a more established and dedicated method of experimentation. The platform is hybrid in that it comprises both real and emulated mobile devices, thus allowing for a diverse and rich ecosystem. Scalability is inherently supported, subject to memory availability and costs related to the purchasing of actual devices. Moreover, the implementation supports flexible definition of scenarios for the experiments as a sequence of steps. Steps refers to particular actions that need to be applied on nodes (real or emulated) of the hybrid experimental platform, e.g. installation of malware app or monitoring of resources. A great diversity of experiments can be conducted utilizing the services of the platform, as for example the estimation of the size of a mobile botnet which was presented in this paper.

The hybrid experimental platform is part of an ongoing exploratory research project and is therefore subject to continuous improvements and extensions of its functionalities. A current limitation involves the need for the users to manually edit XML files to setup configurations of experiments, which can become quite tedious and cumbersome. Accordingly, we plan to develop a GUI to facilitate experimental configurations by providing users with a more interactive visual method. The platform's functionality and operation has been tested, its performance nonetheless has not been evaluated. This is part of our ongoing work, to assess the efficiency of the platform and acquire related quantitative results. We are in the process of expanding our infrastructure to support the integration of more devices and thus have a thorough assessment over the platform's scalability and performance in realistic settings (typical mobile botnets comprise hundreds of nodes).

Lastly, with the aim of detecting mobile botnet attacks and countering them, we plan to explore the potential of supporting hybrid types of experiments with some nodes residing within the scope of the platform interact with actual nodes and botnet malware instances in the wild. The devices in the platform will act as honeypots to detect the operation and behavior of actual mobile botnets and thus assist in their taking down.

REFERENCES

[1] Ahmad Karim, SyedAdeelAli Shah, and Rosli Salleh. Mobile botnet attacks: A thematic taxonomy. In lvaro Rocha, Ana Maria Correia, Felix . B Tan, and Karl . A Stroetmann, editors, *New Perspectives in Information Systems and Technologies, Volume 2*, volume 276 of *Advances in Intelligent Systems and Computing*, pages 153–164. Springer International Publishing, 2014.

[2] A. Malatras, E. Freyssinet, and L. Beslay. Mobile botnets taxonomy and challenges. In *Intelligence and Security Informatics Conference (EISIC), 2015 European*, volume 1, pages 1–4, Sep 2015.

[3] Ruchna Nigam. A timeline of mobile botnets. In *2nd BotConf 2014*, volume 1, pages 1–23, Dec 2014.

[4] Rafael A. Rodríguez-Gómez, Gabriel Maciá-Fernández, and Pedro García-Teodoro. Survey and taxonomy of botnet research through life-cycle. *ACM Comput. Surv.*, 45(4):45:1–45:33, August 2013.

[5] Patrick Traynor, Michael Lin, Machigar Ongtang, Vikhyath Rao, Trent Jaeger, Patrick McDaniel, and Thomas La Porta. On cellular botnets: Measuring the impact of malicious devices on a cellular network core. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 223–234, New York, NY, USA, 2009. ACM.

[6] G. M. Jolly. Explicit estimates from capture-recapture data with both death and immigration-stochastic model. *Biometrika*, 52(1/2):pp. 225–247, 1965.

[7] H.R. Zeidanloo and A.A. Manaf. Botnet command and control mechanisms. In *Computer and Electrical Engineering, 2009. ICCEE '09. Second International Conference on*, volume 1, pages 564–568, Dec 2009.

[8] Brett Stone-Gross, Thorsten Holz, Gianluca Stringhini, and Giovanni Vigna. The underground economy of spam: A botmaster's perspective of coordinating large-scale spam campaigns. In *Proceedings of the 4th USENIX Conference on Large-scale Exploits and Emergent Threats*, LEET'11, pages 4–4, Berkeley, CA, USA, 2011. USENIX Association.

[9] Zonghua Zhang, Ruo Ando, and Youki Kadobayashi. Information security and cryptology. chapter Hardening Botnet by a Rational Botmaster, pages 348–369. Springer-Verlag, Berlin, Heidelberg, 2009.

[10] David Zhao, Issa Traore, Bassam Sayed, Wei Lu, Sherif Saad, Ali Ghorbani, and Dan Garant. Botnet detection based on traffic behavior analysis and flow intervals. *Comput. Secur.*, 39:2–16, November 2013.

[11] Ickin Vural and Hein Venter. Mobile botnet detection using network forensics. In *Proceedings of the Third Future Internet Conference on Future Internet*, FIS'10, pages 57–67, Berlin, Heidelberg, 2010. Springer-Verlag.

[12] Yuanyuan Zeng, Kang G. Shin, and Xin Hu. Design of sms commanded-and-controlled and p2p-structured mobile botnets. In *Proceedings of the Fifth ACM Conference on Security and Privacy in Wireless and Mobile Networks*, WISEC '12, pages 137–148, New York, NY, USA, 2012. ACM.

[13] Jingyu Hua and Kouichi Sakurai. A sms-based mobile botnet using flooding algorithm. In *Proceedings of the 5th IFIP WG 11.2 International Conference on Information Security Theory and Practice: Security and Privacy of Mobile Devices in Wireless Communication*, WISTP'11, pages 264–279, Berlin, Heidelberg, 2011. Springer-Verlag.

[14] Ragib Hasan, Nitesh Saxena, Tzipora Haleviz, Shams Zawoad, and Dustin Rinehart. Sensing-enabled channels for hard-to-detect command and control of mobile devices. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, pages 469–480, New York, NY, USA, 2013. ACM.

[15] Andre Egners, Ulrike Meyer, and Bjorn Marschollek. Messing with android's permission model. In *Proceedings of the 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, TRUSTCOM '12, pages 505–514, Washington, DC, USA, 2012. IEEE Computer Society.

[16] Collin Mulliner, Steffen Liebergeld, and Matthias Lange. Honeydroid-creating a smartphone honeypot. In *Poster session of the IEEE Symposium on Security and Privacy (May 2011), IEEE*, 2011.

[17] Rhiannon Weaver. A probabilistic population study of the conficker-c botnet. In *Proceedings of the 11th International Conference on Passive and Active Measurement*, PAM'10, pages 181–190, Berlin, Heidelberg, 2010. Springer-Verlag.

[18] C. Rossow, D. Andriesse, T. Werner, B. Stone-Gross, D. Plohmann, C.J. Dietrich, and H. Bos. Sok: P2pwned - modeling and evaluating the resilience of peer-to-peer botnets. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 97–111, May 2013.

[19] Tung-Ming Koo and Hung-Chang Chang. Combining the capture-recapture method and simple linear regression analysis of the malicious domains estimation. *Applied Mathematics & Information Sciences*, 7(2L):425–433, 2013.

[20] Cui Xiang, Fang Binxing, Yin Lihua, Liu Xiaoyi, and Zang Tianning. Andbot: Towards advanced mobile botnets. In *Proceedings of the 4th USENIX Conference on Large-scale Exploits and Emergent Threats*, LEET'11, pages 11–11, Berkeley, CA, USA, 2011. USENIX Association.

[21] C. Mulliner and J.-P. Seifert. Rise of the ibots: Owning a telco network. In *Malicious and Unwanted Software (MALWARE), 2010 5th International Conference on*, pages 71–80, Oct 2010.

[22] Jon Oberheide and Farnam Jahanian. When mobile is harder than fixed (and vice versa): Demystifying security challenges in mobile environments. In *Proceedings of the Eleventh Workshop on Mobile Computing Systems &#38; Applications*, HotMobile '10, pages 43–48, New York, NY, USA, 2010. ACM.

[23] Erol Gelenbe, Gke Grbil, Dimitrios Tzovaras, Steffen Liebergeld, David Garcia, Madalina Baltatu, and George Lyberopoulos. Nemesys: Enhanced network security for seamless service provisioning in the smart mobile ecosystem. In Erol Gelenbe and Ricardo Lent, editors, *Information Sciences and Systems 2013*, volume 264 of *Lecture Notes in Electrical Engineering*, pages 369–378. Springer International Publishing, 2013.

[24] Matthias Wählisch, Sebastian Trapp, Christian Keil, Jochen Schönfelder, Thomas C. schmidt, and Jochen Schiller. First insights from a mobile honeypot. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '12, pages 305–306, New York, NY, USA, 2012. ACM.

[25] Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. Planetlab: An overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12, July 2003.

[26] András Varga et al. The omnet++ discrete event simulation system. In *Proceedings of the European simulation multiconference (ESM2001)*, volume 9, page 65. sn, 2001.