Mobile Botnets Development: Issues and Solutions

Paolo Farina, Enrico Cambiaso, Gianluca Papaleo, and Maurizio Aiello

Abstract—**Due to their limited capabilities, mobile devices have rarely been adopted as attack vectors. In this paper, we consider the execution of coordinated and distributed attacks perpetrated by mobile devices (mobile botnet). We first describe current botnets architectures, analyzing their strengths and weaknesses. Then, we identify problems deriving from the development of a mobile botnet. Appropriate solutions to such problems have been proposed, thus providing an important resource during design and development stages of a mobile botnet.**

Index Terms—**Distributed attacks, network attacks, network security, system architecture, smartphones, denial of service.**

I. Introduction

The world of Internet is in constant expansion and more and more people, groups, companies, and governments depend on the network to effectively perform a wide range of activities.

Recently, thanks to the development and diffusion of mobile devices, such as the so-called smartphones, users can easily access to the Internet using access points distributed along the territory and through different technologies of network types (Wi-Fi, 3G, LTE, etc…). Because of this wide adoption, these devices have become an important and crucial element of people lives.

Initially, hardware on mobile devices was not even remotely comparable to those of a real computer. Today, instead, mobile phones are able to execute most operations and tasks normally performed by a personal computer (i.e. sending/receiving e-mails, web browsing, processing and management of multimedia contents, file sharing, etc…). At the same time, the expansion and extension of mobile networks, together with their progressive integration and interconnection with the Internet, allows users to increase the ubiquity of access points with an ever growing increase of the available bandwidth, also thanks to technologies such as LTE [1], which (theoretically) allow connections at speeds higher than those available on an ADSL home link [2].

As a consequence, the cybercrime world quickly recognized the potentialities of mobile devices, developing specific malicious software such as worm or spyware [3]. Then, in the face of this phenomenon, cyber-security has had to deal with new and unknown issues relative to the nature of the mobile environment, with particular attention to privacy and sensitive data protection.

In the arena of attacks exploiting network nodes, *botnets* [4] represent an important resource for cybercrime organizations. In particular, a botnet is a network of infected hosts (known as *bots*, *agents*, or *zombies*) under the control of a *bot*-master that can send specific attack commands to the bots in order to carry out distributed and coordinated operations [5], typically (but not only) aimed at launching cyber-attacks. Recently, such menaces have been adapted to the mobile environment (i.e. iKee.b iPhone botnet [6]).

It may be evident that, given their huge and increasing number, mobile devices could constitute not only a target of attacks, but also a particularly effective resource for executing distributed attacks. We talk in this case of a *mobile botnet*. Since mobile resources are limited, attacks that make a lightweight use of them are preferable. For instance, due to the limited bandwidth requirements, Slow DoS Attacks (SDA) are particularly suitable to a mobile execution [7].

The rest of the paper is structured as follows: Section II describes how distributed attacks and classical botnets work. Section III analyzes different botnet architectures. Instead, Section IV introduces problems related to mobile botnets, while Section V presents appropriate solutions and mechanisms for the creation of an efficient and optimized botnet, giving special attention to the mobile environment. Finally, Section VI reports the conclusions of the work.

II. Classical Botnets

As anticipated above, a botnet is a network of infected hosts (under the control of a bot-master) with the purpose to launch distributed attacks taking advantage of the resources offered by many hosts, for instance for password cracking or decryption purposes [9].

A botnet is mainly characterized by its *command and control infrastructure* (*C&C*), used by the bot-master to communicate its orders to the agents. Also, a client program, installed on every bot, is used to communicate with the C&C infrastructure, in order to receive and execute the attack commands.

Before creating a botnet, a preliminary agents recruitment stage is accomplished, typically silently trying to distribute the client software on a wide set of machines, infecting them using the same techniques employed for worms spreading over the Internet.

Next, an agent coordination and control phase is performed, propagating commands to the bots and consequently collecting botnet information and responses.

We will now describe in detail these two phases.

A. Agents Recruitment

The agents recruitment phase may be accomplished in a *direct* or *indirect* way: a direct agents recruitment action

Manuscript received August 22, 2014; revised October 24, 2014.

The authors are with National Research Council, Genoa, Italy (e-mail: paolo.farina@ieiit.cnr.it, enrico.cambiaso@ieiit.cnr.it, gianluca.papaleo@ieiit.cnr.it, maurizio.aiello@ieiit.cnr.it).

typically occurs in three distinct stages:

- The first stage consists in identifying a "sufficient" number of machines in virtue of their exposition to penetration attacks;
- Then, the attacker takes control of the previously identified machines, by exploiting one or more vulnerabilities afflicting them [10];
- Finally, appropriate software will be installed in a hidden way on the exploited machines, with the purpose of using it for controlling and coordinating attacks.

These three stages can be performed in a manual, semi-automated or fully automated way.

Instead, an indirect agents recruitment action makes use for instance of an Internet worm or a spam/phishing campaign. In particular, in order to create a botnet composed by a high quantity of agents, due to their spreading capabilities, worms adoption may result the better choice [15, 16].

B. Agents Coordination and Control

After reaching a sufficiently high number of agents, the attacker would try to communicate with them by using a one-to-many communicative approach. In particular, the attacker typically requests the following operation executions:

- Start or stop an attack;
- Specify the attack type and targets;
- Obtain information relatively to the status of the botnet and the bots;
- Update the communication system and/or the attack software installed on the agents.

III. Botnet Topologies

Different network topologies may be adopted by a botnet, depending on the needs of the attacker. We will now report in detail classical architectures used for command and control activities.

A. Centralized Models

The centralized model is the simplest one and it is characterized by a central point (sometimes called *handler*) forwarding control messages to the agents.

We will now describe different approaches that may be applied in this context.

1) Direct communication

In this communication model, commands are sent by the attacker to a machine/process called handler, which will then forward the received orders to the active agents of the botnet. The handler also takes care of updating the list of active bots, identifying them from their IP address, a parameter that may change over time.

Botnet control systems often implement authentication mechanisms to access the handler in order to avoid third parties sending orders to the agents or eventually interrupting a running attack.

Benefits**:**

- Operations and administration procedures are simplified.

Drawbacks**:**

- Easily traceable structure: since each agent knows the handler's identity, by capturing an agent machine it is possible to identify the handler, hence being able to identify all the other agents of the botnet, thus dismantling the entire network;
- Easy detection of the botnet: anomalies may easily be detected by analyzing and monitoring network logs (i.e. the opening of a TCP suspect port on the handler machine);
- Management of a limited number of agents (typically not greater than a thousand), due to the limited resources of the process handler.

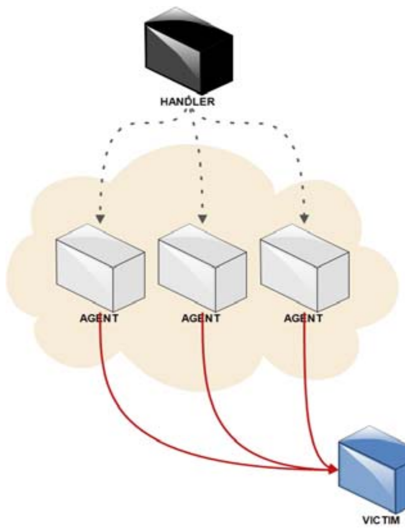

Fig. 1. Command and control model based on direct communication.

Fig. 1. reports an example of direct communication between a machine (the handler), which broadcasts the commands given by the attacker, and the agents who carry out the attack [11].

2) IRC-based communication

Created by Jarkko Oikarinen in August 1988, Internet Relay Chat (IRC) is a client-server protocol that represents one of the first instant messaging communication systems on the Internet. It is described in RFC 1459 and RFC 2810 in its most recent IRC version 2 [12].

IRC is a plain-text protocol using TCP transport layer protocol and supporting TLS. The server is able to connect to other similar IRC servers, in order to extend the communication network.

Communication between users is accomplished on the so-called *channels* (sometimes also called *rooms*), identified by the server with a unique name. The first user joining a channel is considered the *channel operator*. A set of privileges is assigned to such user, allowing him to moderate and take control of the channel.

Since IRC connections are usually not encrypted and they remain active for long times, the protocol has always attracted hackers and malicious users. Because of this reason, historically, IRC systems have been victim of several types of attacks. Contemporary, IRC has often represented a communication tool for the creation and exchange of offensive software and illegal contents.

Relatively to command and control systems, the IRC protocol is often used for communications, since it guarantees good anonymity and reliability levels [13]. In this case the agents and the bot-master connect to a common

IRC server in order to communicate through a predefined channel.

Generally, at start-up, the bot program always connects to a specific IRC channel, whose address is hardcoded in the software. Through this channel, a secondary real control room actually used by the botnet is specified to the agents, in order to make them join it. In addition, a channel change can be induced at any time, through appropriate commands. For instance, a technique called *channel hopping* consisting on periodic and automatic channel changes may be adopted to increase evasiveness.

Due to the distributed nature of the IRC protocol, it is not necessary that all agents are connected to the same server to be part of the botnet. Indeed, it is sufficient to reach a server on the same IRC network: in this way a channel can be virtually distributed among multiple servers, thus augmenting system's reliability and scalability.

Also, in order to improve botnet reliability and to better obfuscate the communication network, attackers often tend to compromise some hosts, thus installing customized IRC servers listening on non-standard ports. In addition, a so-called *stepping stone machine* may be used by the attacker as an intermediary for commands sending.

Examples of IRC based botnets are Agabot, SDBot, and Spybot [14].

Benefits:

- Detection of control traffic is particularly difficult: anomalies are normally not generated, since it is needed to analyze thousands of IRC channels normally used by legitimate users;
- High level of anonymity and evasiveness through the use of the stepping stone, a particularly complex network structure, and channel hopping technique;
- Scalable infrastructure that can handle a significantly large number of agents (over 500,000), with additional IRC servers.

Drawbacks:

- Known vulnerabilities and attacks affecting IRC systems may be used to destabilize or dismantle the botnet.

3) Communication through HTTP and Fast-Flux Networks

The exploitation of the HTTP protocol in botnet's C&C systems has gradually spread thanks to its characteristics of evasiveness and easiness of use and deployment.

For instance, BlackEnergy is a botnet based on the HTTP protocol and designed to perform Distributed Denial of Service (DDoS) attacks.

In general, the functioning of these HTTP exploiting systems is based on the use of a web server contacted by the bots through regular HTTP requests.

In this context, it is also important to mention the so-called *fast-flux technique* [15], [16], adopted to hide and anonymize the machines that are part of a botnet through the DNS protocol. In particular a fast-flux network can be adopted to:

- Use multiple continuously changing IP addresses, assigned to the same DNS domain name;
- Generate a large number of DNS domains pointing to the same IP address;
- Use multiple DNS domains pointing to multiple IP addresses, continuously changing the IP/DNS association.

The idea of fast-flux networks essentially consists in the obfuscation of the IP address associated to the C&C infrastructure using the DNS protocol. In this way, each bot of the botnet is still able to communicate with the C&C system using different domains/IP addresses and the chance to be detected is reduced.

These techniques are also usually combined and adopted to obstacle communications tracing executed to reconstruct the botnet structure, for example during post-incident/forensic analysis.

Benefits:

- botnet detection and tracing operations are hindered: anomalies are normally not generated due to the use of a commons protocol such as HTTP;
- high level of anonymity and evasiveness may be reached thanks to fast-flux service networks and proxy systems.

Drawbacks:

- the high complexity of the network could slow down bots responsiveness during the attack operations.

B. P2P Based Models

Recently, models based on peer-to-peer (P2P) networks appeared [16], [17]. A peer-to-peer system is a network composed by nodes that can act both as a client and as a server. In particular, in such models communications spread through the P2P network itself.

Due to their intrinsic characteristics of reliability and robustness, P2P based networks are often used as botnet architectures, since it is particularly difficult to detect and dismantling them [18]-[20]. For instance, the Phatbot botnet adopts a P2P based communication system that makes use of the WASTE protocol [21] and connects the agents using a Gnutella caching server [22].

Other known botnets using this model are Storm [23], [24], SpamThru, Nugache, and Peacomm (based on Kademlia) [23].

Benefits:

- Very high level of anonymity and reliability inherited from the P2P network structure.

Drawbacks:

- Network address translation (NAT) systems and firewalls can obstacle botnet communications: for example, a peer behind a NAT (without a public IP address) could not act as a server and receive inbound connections coming from other peers.

C. Not-Structured Models

In this type of model bots don't actively contact other entities of the botnet. Instead, they run a service listening on a specific port. When the attacker wants to communicate with them it will search for bots randomly scanning a network (e.g. all active hosts in an IP addresses range) delivering the intended message to each BOT found (Table I).

Another mechanism could consist in some sort of bots search in a network. Once a bot is found, a message is delivered to him by the attacker, and the bot would itself look for other agents on the network, with the purpose of communications dissemination. In this case, the bot-master don't exactly know how many and which bots are part of the botnet, but he's able to communicate with them.

TABLE I: SUMMARY TABLE DESCRIBING THE CHARACTERISTICS OF DIFFERENT BOTNET TOPOLOGIES

Topology	Complexity	Evasiveness	Responsiveness	Reliability
Centralized	Low	Medium	Low	Low
Peer-to-Peer	Medium/High	Low	Medium	Medium
Non-Structured	Low	High	High	High

An example of botnet implementing such model is Sinit [22].

Benefits:

- Simple implementation of communication mechanisms.

Drawbacks**:**

- The bot-master can have only poor information about the botnet infrastructure and bots;
- Firewall and NAT systems can hide bots making them potentially unusable due to the impossibility of receiving inbound connection;
- Bots discovery may take long times due to the IP addresses scan.

Table I summarizes the characteristics of the described botnet topologies [25], in terms of complexity of the network, evasiveness to Intrusion Detection Systems, responsiveness in communications, and reliability of the network.

It is also important to consider that, in order to maintain the offensive potential of the botnet during the time, it is essential to provide mechanisms for updating the bot software installed on the agents. Such updates are needed in order to support additional attacks, improve the effectiveness of the already implemented ones, provide additional detection and mitigation circumnavigation techniques, or just to fix software bugs. This mechanism is usually implemented by sending an update command to the agents, thus making them download the new version of the software, for instance from an HTTP or FTP server.

In case the botnet is structured according to a P2P model, updates could easily be transmitted via the network itself through a node-to-node communication, without requiring external elements.

IV. MOBILE BOTNETS ISSUES

Due to the recent large-scale diffusion of mobile devices, phenomena until now bounded to desktop computing are moving in direction of portable systems [26]. In particular, we believe that botnet development will *even* more involve mobile devices. In this area, it is crucial to face with a new series of problems occurring when classical botnet structures are brougth in the mobile environment.

A. Low Computational and Storage Resources

Although the gap between mobile and desktop hardware is even thinner, computational and storage resources on mobile devices still have to be considered as a limit. Indeed, heavy operations (such as the execution of particularly resource intensive distributed attacks) have to be considered un-deployable. For example, let's think about a botnet used to create an anonymous file-sharing network: in this case, the possibility of running out of disk space on mobiles is *particularly* high, thus making the sharing software ineffective.

Furthermore, in order to provide good levels of evasiveness and anonymity to the bot-master, many structures (such as the IRC based ones) make use of some bots as intermediate elements to deliver commands directed to the agents. These intermediaries may also execute complex command and control functions relatively to the management of a specific set of agents. Since such operations are particularly resource intensive, this role shouldn't be assigned to a mobile device.

B. Limited Network Bandwidth

Another issue related to the mobile environment is linked to limitations on traffic, often imposed by Internet service providers. Since classical botnet infrastructures are frequently designed to involve always connected hosts, many systems may experience problems. For instance, botnet infrastructures using known and sophisticated protocols and encryption systems may experience high overheads and generate high traffic volumes during the propagation of messages on the network.

Indeed, analyzing the types of attacks usually executed by a botnet, it is important to observe that some categories of attacks (e.g. flooding DoS attacks [27]) require the perpetrators to heavily use the available resources in order to be effective. Therefore, in this case a mobile device may quickly consume the available bandwidth. Other attacks, such as Slow DoS Attacks [7], require instead extremely low amount of bandwidth.

C. Mobile Network and Connection Characteristics

During the design phase of a mobile distributed system it is important to consider cellular network evolutions and their characteristics: communications mechanisms for data exchange should be considered with particular attention. For instance, a *proxy accelerator* placed on the outgoing network path (with the purpose of speeding up *communications*) may obstacle botnet communications and operations, being it based on intermediation mechanisms (caching, data compression, etc…) that can alter the original informative data flux [28]-[30].

Furthermore, due to mobility, devices continually and quickly change their connection endpoint, even through heterogeneous networks. By changing the *point of attachment* (*POA*) through a *vertical/horizontal handover*, each bot may be located in a different network environment, thus being subject to different restrictions.

For *example*, an IRC C&C mechanism may cause continuous connections closures due to frequent handovers, since a persistent connection with the IRC server is needed. In this case, since the number of active bots on the mobile botnet may be extremely dynamic, the bot-master should be aware that the subset of active bots is constantly changing. Frequent losses of connectivity may also derive from the use of wireless connections technologies: these links are intrinsically unreliable, due to possible interference and attenuation phenomena.

D. Energy Consumption

Since mobile devices are almost always powered by batteries, *bot* software may result ineffective in case its operations bring to anomalous energy consumptions (battery drain). As a result of this inefficiency, users may be induced to reset the device firmware or even to replace the device itself. In general, power consumption depends on resources usage, including CPU, memory, bandwidth, and storage.

Although this issue is not particularly relevant for "wired powered" devices (like personal/desktop computers), it assumes importance on mobile devices with limited power capabilities.

V. Proposed Solutions

During the developing process of a mobile botnet, the problems introduced in the previous section have to be solved in order to provide an effective and reliable attack distribution system. We will now propose solutions to these problems.

A. Optimized Software and Infrastructure

In order to reduce software consumption in terms of needed resources, some optimization strategies may be applied. In conjunction with software development cycles [31], three characteristics should particularly be considered:

- Good design and cost estimation: during the design stage, a developer has to assess every design choice, in order to reduce software size and optimize system infrastructure. In general, we believe that it is fundamental to reduce to a minimum the features implemented on the client, delegating any intensive task to an external server. In particular, it is extremely difficult to deploy a botnet only composed by mobile devices. Nevertheless, a hybrid botnet composed by both mobile and wired powered bots may adopt both centralized and hierarchical architectures, thus guaranteeing good responsiveness and reliability characteristics.
- High-quality code: continuous code optimization and refactoring activities are crucial, in order to enhance the performance of the software, simplify extensions operations, and reduce potential bugs. Moreover, by choosing efficient algorithms, CPU consumptions are reduced. It is also important to carefully evaluate the use of external (third-party developed) libraries or modules, since it may expose the software to additional bugs or overheads caused by non-optimized operations.
- Performance testing with configuration adjustments: the final development stage consists in the execution of accurate performance testing sessions, accomplished to evaluate the system behavior in a real environment, with the purpose of adjusting and solving problems and bugs.

B. Simple and Reliable Communication Mechanisms

In a distribution system a fundamental element is represented by the communication between nodes: it is crucial to define and adopt efficient and reliable communication systems.

- Prefer non persistent communication mechanisms: while in many classical botnets communication with agents is based on persistent connections (e.g. IRC based botnets), in case of a botnet composed by mobile agents it is preferable to avoid persistency, due to the unreliability of the communication channel. Therefore, each message has to be sent independently from the others.
- Use simple and known protocols: using common protocols and text formats such as (non persistent) HTTP or XML to implement C&C mechanisms ensures that communications can't be easily thwarted by proxies or firewall systems. We believe that HTTP represents a particularly good solution, as it is commonly used by mobile devices to access Internet services (i.e. web browsing, instant messaging, etc…). Furthermore, by using HTTP, the C&C traffic is indistinguishable from the legitimate one, produced for instance from a web browser. Nevertheless, common protocols may impose an overhead in messages length, while a custom protocol may be designed to optimize the bandwidth.
- Shorten C&C syntax: it is particularly important to simplify and shorten the number of bytes required to send commands to the bots and receive status information from the network.
- Check bots status: although it leads to a considerable communication overhead, it may be very important to propagate accurate information about the bots status, in order to collect useful information for adaptive control mechanisms.

C. Test Server

We will define the Test Server (TS) as a new botnet entity whose role is to provide a status information discovery services to the bot. For example, suppose that we want to implement a botnet to launch DDoS attacks and some network restrictions make a bot unable to carry out the attack, a bot could verify this inability contacting TS. Hence TS should not be implemented using a mobile device and should be connected to the Internet through an unrestricted network, it should be always reachable from the bots. TS is in general necessary when a bot wants to gather information about its connection network. This information are useful to understand its operative possibilities. Indeed, for botnets composed by mobile devices, TS can be used to evaluate the botnet's offensive power.

D. Adaptive Control

For adaptive control, we mean the integration of an intelligent bot management system in the C&C infrastructure. Classical botnets don't implement an intelligent central module and they usually synthetize bot status with two states: *connected* or *disconnected*. Nevertheless, it may be important to extend nodes representation. We introduce the following additional parameters, which may be used to enhance botnet efficiency and evasiveness:

- Connection statistics: it may be interesting to gather Information about connection times and periods, in order to estimate botnet reliability during the time.
- Network information: by using the Test Server and other self testing mechanisms a bot is able to collect information about the network it is connected. In this way, the C&C system directly knows how many and which bots can be effectively used to carry out attacks and actions, with a considerable saving of bots resources.
- Resources usage and status: This information allows the C&C system to manage bot overloading situations. For instance, the C&C system may check bots battery status to redistribute onerous operations to bots with high battery levels.
- Device model and OS characteristics: these parameters

are essential for bot software updates/upgrades propagation. Indeed, since a client software may be developed for different platforms and operating systems, possible updates have to affect specific devices, in order to deliver the correct software package to each bot.

The adaptive infrastructure we propose should be able to dynamically adapt its behavior based on the current situation (connection type, network, device model and platform, botnet functionalities) of the bots, to avoid resource wasting and to increase botnet efficiency, reliability, and evasiveness.

VI. Conclusions and Future Work

Considering the steady increase in the spread of mobile devices, in this article we have faced with the adoption of portable devices as part of coordinated botnet based cyberattacks. We have analyzed the phenomenon, until now bounded to wire powered devices, announcing the possibility of menaces perpetrated by coordinated mobile devices.

We have analyzed possible problems of this porting, first describing different topologies adopted in this context, then introducing issues related to the development of a mobile botnet, and proposing appropriate solutions.

Due to the novelty of the topic, this paper should be considered a first important resource in the mobile botnets field.

Future work will be focused on making use of the proposed solutions to design and develop a mobile botnet.

References

[1] S. Sesia, I. Toufik, and M. Baker, *LTE: The UMTS Long Term Evolution*, Wiley Online Library, 2009.

[2] W. J. Goralski, *ADSL and DSL Technologies*, McGraw-Hill Professional, 2001.

[3] T. F. Stafford and A. Urbaczewski, "Spyware: The ghost in the machine," *Communications of the Association for Information Systems*, 2004, vol. 14, no. 15, pp. 291-306.

[4] A. M. Rajab *et al*., "A multifaceted approach to understanding the botnet phenomenon," in *Proc. the 6th ACM SIGCOMM Conference on Internet Measurement*, 2006, ACM.

[5] Z. Zhaosheng *et al*., "Botnet research survey," in *Proc. 32nd Annual IEEE International Computer Software and Applications*, pp. 289-304, 2008.

[6] P. Porras, H. Saidi, and V. Yegneswaran, "An analysis of the ikee. b iphone botnet," *Security and Privacy in Mobile Information and Communication Systems*, 2010, Springer, pp. 141-152.

[7] E. Cambiaso *et al*., "Slow DoS attacks: definition and categorisation," *International Journal of Trust Management in Computing and Communications*, 2013, vol. 1, no. 3, pp. 300-319.

[8] J. Mirkovic, *Internet Denial of Service: Attack and Defense Mechanisms*, 2005, Prentice Hall.

[9] D. Seeley, "Password cracking: a game of wits," *Communications of the ACM*, 1989, vol. 32, no. 6, pp. 700-703.

[10] J. Erickson, *Hacking: The Art of Exploitation*, 2008, No Starch Pr.

[11] A. K. Pandey. Survey on Attacks targeting Web Based System Through Application Layer. [Online]. Available: http://medianet.kent.edu/surveys/DR05S-applicationattack amitpandey/index.html.

[12] C. Dewes, A. Wichmann, and A. Feldmann, "An analysis of internet chat systems," in *Proc. the 3rd ACM SIGCOMM Conference on Internet Measurement*, 2003, pp. 1-22.

[13] Internet Relay Chat. [Online]. Available: http://www. en.wikipedia.org/wiki/Internet_Relay_Chat.

[14] P. Barford and V. Yegneswaran, "An inside look at botnets," *Malware Detection*, 2007, Springer, pp. 171-191.

[15] J. Riden, "Know your enemy: fast-flux service networks," The Honeynet Project, 2008.

[16] R. Schollmeier, "A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications," in *Proc. First International Conference on the Peer-to-Peer Computing,* 2001.

[17] R. Steinmetz, *Peer-to-Peer Systems and Applications*, Springer-Verlag New York Incorporated, vol. 3485, 2005,

[18] H. Binsalleeh *et al.*, "On the analysis of the zeus botnet crimeware toolkit," in *Proc. the International Conference on Privacy Security and Trust (PST)*, pp. 1-10, 2010.

[19] D. Dittrich and S. Dietrich, "P2P as botnet command and control: a deeper insight," in *Proc. 3rd International Conference on Malicious and Unwanted Software,* 2008.

[20] D. Sun *et al*., "The new architecture of P2P-botnet," presented at Cybercrime and Trustworthy Computing Workshop (CTC), 2010,

[21] C. Li, W. Jiang, and X. Zou, "Botnet: survey and case study," in *Proc. the Fourth International Conference on Innovative Computing, Information And Control (ICICIC)*, vol. 129, 2009.

[22] P. Wang, S. Sparks, and C. C. Zou, "An advanced hybrid peer-to-peer botnet," *IEEE Transactions on Dependable and Secure Computing*, vol. 7, no. 2, pp. 113-127.

[23] J. B. Grizzard *et al*., "Peer-to-peer botnets: overview and case study," in *Proc. the First Conference on First Workshop on Hot Topics in Understanding Botnets*, 2007, pp. 1-8.

[24] T. Holz *et al*., "Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm," in *Proc. the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, 2008, San Francisco, California, pp. 1-9.

[25] E. Cooke, F. Jahanian, and D. McPherson, "The zombie roundup: Understanding, detecting, and disrupting botnets," in *Proc. the Usenix Sruti Workshop*, 2005, pp. 1-6.

[26] A. Flo and A. Josang, "Consequences of botnets spreading to mobile devices," in *Proc. the 14th Nordic Conference on Secure IT Systems (NordSec 2009)*, 2009, pp. 264-279.

[27] J. Yuan, and K. Mills, "Monitoring the macroscopic effect of DDoS flooding attacks," *IEEE Transactions on Dependable and Secure Computing*, 2005, vol. 2, no. 4, pp. 324-335.

[28] H. Bharadvaj, A. Joshi, and S. Auephanwiriyakul, "An active transcoding proxy to support mobile web access," in *Proc. the Seventeenth IEEE Symposium on Reliable Distributed Systems*, 1998, pp. 2-6.

[29] A. Feldmann *et al*., "Performance of web proxy caching in heterogeneous bandwidth environments," in *Proc. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, Proceeding*, 1999, pp. 1-10.

[30] D. Rosu, A. Iyengar, and D. Dias, "Hint-based acceleration of Web proxy caches," *IEEE International Performance*, vol. 12, 2000.

[31] R. S. Pressman and W. S. Jawadekar, *Software Engineering*, New York, 1987.

[32] A. P. Felt *et al*., "A survey of mobile malware in the wild," in *Proc. the 1st ACM workshop on Security and Privacy in Smartphones and Mobile Devices*, vol. 4, 2011.

[33] M. Becher *et al*., "Mobile security catching up? revealing the nuts and bolts of the security of mobile devices," in *Proc. IEEE Symposium on the Security and Privacy (SP)*, 2011, pp. 96-111.

Paolo Farina was graduated in computer science at the University of Genova in 2013, with a thesis entitled "Analysis and Development of a Cooperative Network for 'Slow DoS' Attacks Using Mobile Devices". Nowadays he is a research fellow at CNR-IEIIT.

Enrico Cambiaso was graduated in computer science at the University of Genoa in 2012, with a thesis entitled "Analysis of Slow DoS Attacks". He is a PhD student at the University of Genoa and he collaborates with CNR-IEIIT, working to the slow DoS field.

Gianluca Papaleo was graduated in computer science at the University of Genova in 2005. His fields of research are network security, intrusion detection systems, covert channel, DoS. Since 2006 he is a research fellow at CNR-IEIIT.

Maurizio Aiello was graduated in 1994 and he worked as a free-lance consultant in the field of system and network management both for universities and research center and for private industries. From August 2001, he is responsible for the Region of Liguria of National Research Council network infrastructure. He is a teacher of the course 'Network Security' at the University of Genoa and University College of Dublin. he is also a coordinator of students, fellowships and EU projects in the field of computer security. He is involved in activities related to technology transfer (spin-off). His research activities are network security and protocols.