Insight into Contemporary Dissemination Techniques of Mobile Botnet Clients (Bots)

Milan Oulehla
Faculty of Applied Informatics
Tomas Bata University in Zlin
Zlin, Czech Republic
oulehla@fai.utb.cz

David Malanik
Faculty of Applied Informatics
Tomas Bata University in Zlin
Zlin, Czech Republic
dmalanik@fai.utb.cz

Abstract— **Currently, smartphones and tablets offer a wide range of functionalities, such as Web browsing, social networking, as well as using banking applications. This has resulted in s constant increase in popularity of mobile devices connected to the Internet 24/7. In the 2nd quarter of 2015, the Android operating system has dominated the market with an 82.8% share, which makes it the most widespread mobile operating system in the world. However, this popularity is double-edged, including both users and botnet creators. The research papers "Android Botnets on the Rise: Trends and Characteristics and How Can Botnets Cause Storms?" as well as "Understanding the Evolution and Impact of Mobile Botnets" imply urgent need for additional research into this field. Therefore, the research described by this article includes not only a theoretical study into the ways of delivering the botnet command and essential botnet knowledge based on published research, but also provides a practical investigation into the ways of infecting smartphones and tablets with botnet clients (bots). Special tools have been developed for performing certain malicious actions enabling real testing of safety mechanisms of the Google Play. These tools have also been used in combination with other useful techniques such as social engineering and deceitful actions trying to get users to unintentional cooperation. Finally, some challenging results and security vulnerabilities have been raised by the research.**

Keywords- Android permission analysis; bot (client of botnet); bot dissemination, C&C server; Google Play; mobile botnet

I. INTRODUCTION

Mobile devices, such as smartphones, tablets and wearable devices are able to perform a whole range of features resulting in their utilization for both personal purposes, such as Web browsing, social networking, using banking applications, etc. [11][10], and for business purposes, including continuous access to corporate mailbox and real-time file sharing [11]. Besides these functionalities, most of the mobile devices are connected to the Internet 24/7 and unlike personal computers they can use different connections to the Internet using technologies such as EDGE[1], 3G[2], HSDPA[3], Wi-Fi[4], etc. [12]. All these factors cause continuous growth in the popularity of mobile devices. This research focuses on Android because it is one of the most popular mobile operating systems in the world, which had over 1 billion active users in 2014 [24]. One year later, in the 2nd quarter of 2015, the Android operating system had 82.8% market share [14]. The research papers "Android Botnets on the Rise: Trends and Characteristics" [20] and "How Can Botnets Cause Storms? Understanding the Evolution and Impact of Mobile Botnets" [25] suggest that popularity of mobile devices is double-edged, including both users and botnet creators. The findings published in [21] are alarming. The researchers examined 1,632 popular applications published on Google Play. They employed methods of static analysis, which contribute to the revelation that 151 applications represent a potential security threat. The investigation published in [15] has also brought concerning results: 93% out of 1,260 tested mobile malware samples contained patterns of botnet behavior. Such situation is as serious as the one similar to mobile antivirus field. In [19], a prototype of hybrid command and control mobile botnet has been created and subsequently tested by four mobile antivirus programs with worrying results: "All the anti-viruses were active during the execution of the prototype but failed to identify any malicious activities". All facts stated above imply the urge for further research into the field of mobile botnets. This paper contributes to the mobile platform security improvement.

[1] Enhanced Data rates for GSM Evolution (EDGE)

II. INSIGHT INTO BOT DISTRIBUTION ISSUES

A. Principal terms

In order to better understand mobile bot distribution issues, with emphasis on the Android platform, seems useful to introduce some main terms used in this field.

- Google Play is a software distribution platform for mobile devices. Google Inc. has developed an automated antivirus system, called Google Bouncer with the purpose of finding and removing malicious software published on Google Play [16].

[2] third generation of mobile telecommunications technology
[3] High-Speed Downlink Packet Access (HSDPA)
[4] wireless local area network

 ISBN: 978-1-61208-493-0

- Bot (also known as agent or zombie) is a piece of malicious software installed on the mobile devices of victims [11]. Bots are clients of botnet network and botmaster can control them via C&C server[5] [3].
- AndroidManifest.xml is a file, which is an inseparable part of every Android application. "The manifest file presents essential information about a described app to the Android system; the system requires this information before it can run any of the app's code. It describes the components of the application including activities, services, broadcast receivers, and content providers as well as declares, which permissions the application must have in order to access protected parts of the API[6] and interact with other applications." [6]
- Dynamic application analysis is concentrated on application patterns of behavior. Inspected applications are executed in controlled environment and their running is logged and subsequently evaluated (e.g., analysis of captured *.pcap files). Methods of dynamic analysis try to find certain anomalies in network traffic, battery consumption, CPU[7] utilization, etc.
- Static application analysis, unlike dynamic application analysis, focuses on inspection of the source code. "An application is analyzed without its executing." [1]. It typically consists of a decompilation phase and a code analysis. The tools including Dex2Jar [1], JD-GUI [1], Apktool [13] and Virtuous Ten Studio (VTS) [13] are employed during the process of static analysis. In the cases of Smali, Java and XML code analysis, pieces of malicious code are searched for. Nevertheless, there are certain techniques, which make the accurate automatic code analysis more difficult for example code obfuscation or harmful intention camouflaged by a programming style. It is also quite difficult to programmatically decide whether it is malicious intent or just badly written part of code. All these pitfalls result in the fact that the static analysis methods cannot be easily converted to the automated code analyzer.
- BroadcastReceiver [9] is an Android Java class, which does not have any user interface and therefore it can run silently in the background. From this place it processes events from the system or other applications including: SMS[8] has been received, a device is connected via Wi-Fi, certain custom application events, etc. All these features make BroadcastReceiver convenient for performing harmful actions of mobile malware. Google, the creator of Android operating system has realized threats resulting in using of BroadcastReceivers. For this reason starting from Android 3.1 and higher, every application, which wants to use BroadcastReceiver requiring certain permissions also has to have an Activity [7]. This measure caused that malware creators have focused on techniques allowing camouflage of malware Activities.
- "PUSH" is the way of botnet command delivering, during which commands are sent from the C&C server to bots [12].
- "PULL" is the way of botnet command delivering, during, which bots periodically send requests to the C&C server. Then server sends commands as responses [12].

[5] Command-and-control server
[6] Application Programming Interface
[7] Central processing unit
[8] Short message service

B. *Android permission analysis*

Google Play as well as anti-viruses analyze permissions from AndroidManifest.xml file [20]. From the Android applications' point of view, there are two kinds of permissions: function permissions and actually requested permissions (see Figure 1, set A and B). However, there is another point of view, which is represented by Android system permissions [8]. From this perspective, normal and dangerous permissions exist (see Figure 1, set C and D) [8]. Function permissions represent a group of permissions which are legitimate. Function permissions represent a group of permissions which are legitimate because this application is not able to perform its function without RECORD_AUDIO permission. set of all permissions an application asks for is called requested permissions. It can contain both functional (legitimate) and illegitimate requests. Normal permissions form a static list of permissions, which are not considered to be dangerous. It also surprisingly contains permissions such as INTERNET, ACCESS_NETWORK_STATE and RECEIVE_BOOT_COMPLETED.

These permissions are particularly suitable for communication with C&C servers. Dangerous permissions consist of permissions by, which serious harm could be caused e.g., RECORD_AUDIO or READ_CONTACTS and more. Because of this, the set of functional and requested permissions differs in most applications. Compared to the sets of normal and dangerous permissions, which are static and always have the same elements. The system of dynamic and static permission sets helps to improve accuracy of Android permission analysis. The analysis process tries to find permission discrepancies, which could be expressed as: $A - B \neq \emptyset$. According to [8] the emphasis is put on: $(A - B) \cap D \neq \emptyset$. For example, a real voice recording application could ask for functional permission such as above-mentioned RECORD_AUDIO, however, it can request for any Android permission e.g., READ_CONTACTS, RECEIVE_BOOT_COMPLETED and INTERNET. Permissions analyzers try to find the discrepancy between function permissions and requested permissions. Permissions READ_CONTACTS, RECEIVE_BOOT_COMPLETED and INTERNET represent searched contradiction where READ_CONTACTS permission is the most significant security risk. Our preliminary research of Google Play

applications suggests that the phenomenon of excessive permissions is mainly concerns free of charge applications.

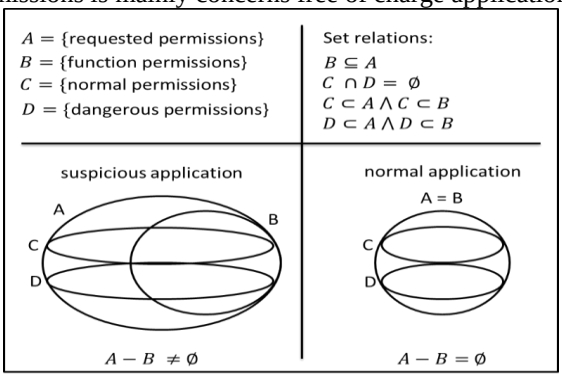

Figure 1. Android permission analysis

III. MEANS OF CONTEMPORARY MOBILE BOT DISSEMINATION

There is a whole range of ways how mobile bots can be distributed into mobile devices:

- Third Party Application Markets are a traditional place where mobile malicious applications have occurred [20]. There is a wide range of mobile malware starting from suspicious applications collecting Web browser history for targeted advertising and ending with sending text messages to premium-rate numbers owned by cyber criminals without user's knowledge [5].
- Android applications, which represent a repackaged version of legitimate applications as were described in [20]. Here is a typical scenario: the paid version of a popular Android game (e.g., Minecraft: Pocket Edition) is decompiled. Then certain malicious code is included. Finally from infected code, APK[9] package is again built. Nevertheless the repackage experiments, which have been carried out during our research suggest that not all repackage APK applications have the same functionality and stability as original APK applications.
- Mobile versions of worms are used for the bot distribution to the mobile device. Worms are able to replicate themselves to other mobile devices using typical security vulnerabilities on the host mobile operating system to infect them [1], [17]. For example, first well known mobile worm was Cabir, which used to spread through Bluetooth technology [26].
- Spam and phishing use sending out emails containing either a hypertext link to infected Web page for downloading bot APK application or an attachment with a bot software, which pretends to be useful in some way [11].
- Malware application for bot dissemination placed on Google Play. As was mentioned earlier, the Android operating system reached over 1 billion active users in 2014 and almost everybody is able to install software from Google Play, except users with inexpensive smartphones or tablets which are not Google certified Android devices. Thus, there is an extremely huge number of potential victims, which transforms Google Play into extremely promising distribution platform for botnet creators. This is probably the most dangerous way of bot spreading because users of Android running devices are used to trust Google Play and they are not alert as during installations from different software sources. Nevertheless, Google Play store has security mechanisms trying to detect and ban malicious applications.

IV. CHARACTERISTICS OF MODERN ANDROID MALWARE

There is a range of factors, which should be taken into account by contemporary malware creators because their ignorance could lead to a disclosure by antivirus programs or by users themselves. There is a list of activities, which well written modern malware should never do:

- Unnecessarily running malware starting from boot of an Android operating system and ending with switching off mobile device since permanent or long-term running of malware results in high battery consumption. Atypical battery consumption can attract user's attention and it can lead to the revelation of malware [19][11].
- Malicious actions demanding high computing power, which could lead to excessive CPU utilization. This phenomenon can be detected by machine learning anomaly detectors [1].
- Sending of stolen data, attack performing (e.g., DDoS attack) and communication (e.g., mobile bot – C&C server) via cellular network using technologies such as EDGE, 3G or LTE. There are several issues: Mobile networks have limited bandwidth and generation of high traffic volumes may quickly consume the available bandwidth [11]. A lot of bots are identified and controlled by IP addresses. Incessant switching between cellular and Wi-Fi network could lead to ambiguous bot identification and subsequent malfunction of botnet as a whole [3].

Communication between mobile malware applications and the servers of attackers is realized via TOR[10] network. Plenty of security scans perform routine network traffic inspection of tested applications and each anonymous communication through TOR network using Onion routing is generally considered suspicious. It could cause employing of more thorough analyses.

V. BOT DISTRIBUTIONAL APPLICATION PUBLISHED ON GOOGLE PLAY STORE

"Malware, packaged within an Android game app called BrainTest, had been published on Google Play twice. Each instance had between 100,000 and 500,000 downloads according to Google Play statistics, reaching an aggregated infection rate of between 200,000 and 1 million users." [22]. These findings are alarming and they imply that despite

[9] Android application package

[10] The Onion Router https://www.torproject.org

Google Bouncer, it is possible to publish application containing malware. In addition, as mentioned above, the research published in [21] has revealed that 9.25% of examined popular applications published on Google Play had a potential security threat. These results have been achieved by employing methods of static analysis. It indicates a hypothesis that Google Bouncer primarily focuses on dynamic analysis and inspection of AndroidManifest.xml file, whereas static analysis is underestimated. This claim is also supported by research, which was carried out by ESET Security Company: "Another interesting issue is why Bouncer didn't statically analyze the executable file inside the assets of the uploaded game. For that reason, the Trojan horse stayed undetected and was freely provided to users." [23]. The target of our research has been influenced by the facts stated above. However, a different approach has been employed. Unlike previously published papers, our research has not concentrated on inspection of existing suspicious applications published on Google Play. In contrast it has been focused on creation of two different bot distributional applications with the same purpose, which should practically prove or disprove a hypothesis about insufficient static analysis of Google Bouncer by using the same scenario:

- the experimental bot distributional applications can be successfully published on Google Play bypassing security mechanisms of Google Bouncer;
- the experimental bot distributional applications can deliver an installation file of bot application to mobile device of victim;
- the experimental bot distributional applications can prepare fraudulent installation of bot application on mobile device of victim.

A. The common basis of experimental bot distributional applications used

Despite the fact that bot distributional applications employing different principles of bot infection, both of them take into account findings of Android permission analysis and characteristics of modern Android malware (discussed in detail above), which can lead to their disclosure. They have also been developed with the purpose to camouflage malicious intentions during dynamic analysis and inspection of AndroidManifest.xml. On the contrary, harmful actions of the experimental applications have been presented in uncovered form in the code of applications, which have even not been obfuscated. It means that both of them have a common base,, which will be described in the following section. The first bot distributional application is called "Spennymoor Weather" (see Figure 2) and the second is called "Meadowfield Weather" (see Figure 4). The target bot application, which should be fraudulently installed by SpennymoorWeather as well as Meadowfield Weather is called "bot application" in this paper. The bot distributional applications have legitimate and illegitimate parts. The legitimate part is an ordinary weather forecast application for Spennymoor and Meadowfield towns. It shows usual meteorological information such as picture of current weather, present temperature, humidity etc. The illegitimate part is designed with emphasis on findings stated in detail above. It makes use of the fact that probably the best form of bot distributional application is a mixture of a bot and a Trojan horse. The illegitimate part has also been created to be able to perform every malicious action only by function permissions of a legitimate part of the application. Due to the lack of IP address, most cellular phones use NAT[11] gateway and thus the devices are not directly reachable [12]. In addition, IP addresses are changed frequently [3] because of incessant switching between cellular and Wi-Fi network. It causes inconvenience in using "PUSH" based communication mechanism. On the other hand, bots employing "PULL" style, regularly establish connection with C&C servers, which could generate additional network traffic. As mentioned in the Insight into bot distribution issues section, anomalies in network traffic can be detected by methods of dynamic analysis. That is the reason why the "PULL" based communication mechanism used by this research has led to the improvement of the mechanism. Illegal part of bot distributional applications has been implemented as background thread with the intention to be operational for as short a time as possible. Unlike classical "PULL" scenario, illegal part of bot distributional applications is not periodically triggered and it does not try to connect to C&C server with request for installation commands. It has been inconspicuously launched by successful downloading of JSON (JavaScript Object Notation), which contains both weather forecast information and control commands for installation of bot application. Once the illegal part is started, a command from JSON is assessed using command evaluate mechanism, which pretends that it is SHA256[12] protection against modification of weatherenginesupportlibrary, an internal component placed in ../res/raw directory of bot distributional application. The weatherenginesupportlibrary is an encrypted array of bytes and thus nobody can investigate its content and purpose. SHA256 value of weatherenginesupportlibrary represents the order for installation of the bot application: The value from downloaded JSON and calculated SHA256 value of weatherenginesupportlibrary are compared. If the values are not equal, it means that command for installation of bot application has not been issued and illegitimate part of application is immediately terminated. Otherwise the process of fraudulent installation continues. The bot distributional applications Spennymoor Weather and Meadowfield Weather employ different methods, which are described separately in the following subsections.

[11] Network Address Translation
[12] Secure Hash Algorithm 256bit

VI. FINDINGS AND RESULTS

A. *Spennymoor Weather – an experimental bot distributional application*

In order to be approved by Google Bouncer, Spennymoor Weather has been designed as inconspicuous as possible. That is the reason why Spennymoor Weather application contains within itself encrypted array of bytes. In fact, it is a bot application, which should be fraudulently installed on the mobile device of a victim. In fact it is an anonymous array of bytes, which was encrypted using AES[13] algorithm with 256-bit key. This measure prevents Google Bouncer from inspecting contents of array and getting results in weakly polynomial time. Spennymoor Weather is a mixture of a bot and a Trojan horse, which is considered to be particularly suitable for bypassing security tests. Since the application has a legitimate purpose and at the same time it is controlled by botmaster, there is no standard observable algorithmic pattern of malicious behavior. Botmaster is a human being whose administration can be quite random. Spennymoor Weather also does not perform any typical malicious actions as memory access violation, gathering/sending users' information or remote code execution. All above mentioned facts resulted in publishing Spennymoor Weather on Google Play store (see Figure 2 and Figure 3/1) so Android users can install it. Once it is installed, the decryption uses a password taken from a variable stored in weather forecast JSON, saves malicious APK in a persistent memory of a device. Subsequently, the fraudulent installation of bot application, which looks like a legitimate update can be performed by order of botmaster (see Figure 3/2). This way, a mobile device can be infected by a bot application, which has not been inspected by any security mechanisms (see Figure 3/3).

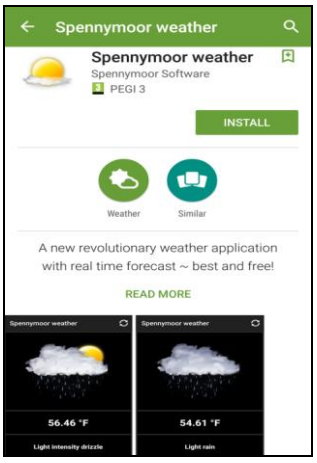

Figure 2. Spennymoor Weather – an experimental bot distributional application on Google Play

[13] Advanced Encryption Standard

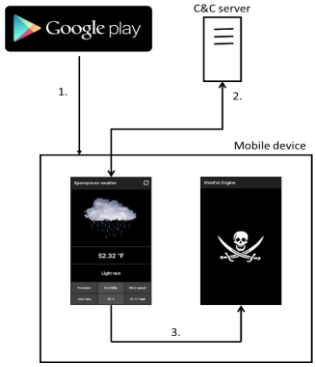

Figure 3. Mechanism of bot dissemination employed by Spennymoor Weather

B. *Meadowfield Weather – an experimental bot distributional application*

The fact that Google Bouncer has allowed publishing Spennymoor Weather containing within itself an encrypted bot application on Google Play, has influenced additional direction of our research in designing Meadowfield Weather application, which is also a mixture of bot and a Trojan horse but has more dangerous features and its malicious intention has been more obvious than in case of Spennymoor Weather application. Despite these facts, Meadowfield Weather has been successfully published on Google Play too, as can be seen from Figure 4 and anybody of more than billion active users is able to download and install it directly from Google Play to her/his mobile device (see Figure 5/1). Moreover it does not include any additional malicious software instead; it is designed to employ two different servers. The first is a C&C server and it is used for controlling bot application installation (see Figure 5/2). The second is a file server, which is designed to offer downloading of bot application to Meadowfield Weather (see Figures 5/3 and 5/4). It means that bot application can be regularly changed according to varying cyber-criminal intentions without the need of code adjusting of Meadowfield Weather. There is a measurement preventing security tests to scan bot application:

- Bot application published on file server is encrypted by AES cipher with complex password [4]. Meadowfield Weather contains fast, single-purpose deciphering subsystem but it does not include a password. This password is sent by a botmaster via C&C server in weather forecast JSON only on condition that an order for installation of bot application was issued. It means that only a combination of Meadowfield Weather deciphering subsystem, encrypted application from file server and password from JSON downloaded from C&C server can lead to decryption of bot application.
- A pair of encrypted bot application and a password can be periodically changed without any code editing of Meadowfield Weather.

 ISBN: 978-1-61208-493-0

- The encrypted application does not have a file extension *.apk in any phase of the installation process, which makes explanation of its purpose more unclear and even if the *.apk file extension is missing; the installation works reliably on Android operating system.

Once the bot application is downloaded on the mobile device, Meadowfield Weather performs decryption using password from weather forecast JSON in a way described above. Then Meadowfield Weather tries to carry out a fraudulent installation employing a method of social engineering by which it attempts to persuade the user to finish this installation of bot application (see Figure 5/5). Meadowfield Weather pretends that the bot installation process is an update of Weather Engine, which is necessary for operation of Meadowfield Weather. The moment the bot application is installed on the mobile device, there is a whole range of techniques, which could be used for its camouflage e.g., [18].

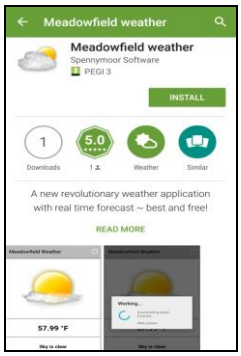

Figure 4. Meadowfield Weather – an experimental bot distributional application on Google Play

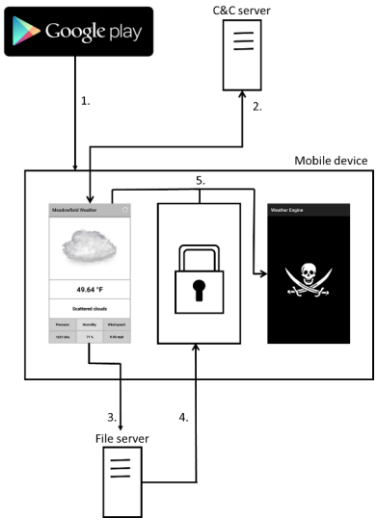

Figure 5. Mechanism of bot dissemination employed by Meadowfield Weather

VII. CONCLUSION

This article deals with dissemination techniques of modern mobile bots. Due to better understanding to the research described in this paper, certain main terms have been explained primarily. Means of contemporary mobile bots dissemination have been described on the basis of preliminary research carried out and published papers in the corresponding field. The study of Android permission analysis and characteristics of modern Android malware have enabled to design Spennymoor Weather and Meadowfield Weather applications. These two pieces of experimental bot distributional software represent a mixture of a bot and a Trojan horse and they have been created as inconspicuously as possible. They have a legitimate part, which is a weather forecast application for Spennymoor and Meadowfield towns. The illegal part does not perform any typical malicious actions as is gathering sensitive personal information or periodical connecting to C&C server. It has only one purpose, which is a fraudulent installation of bot application. These features resulted in a fact that both of them have been able to bypass Google Bouncer security mechanisms. Moreover, they enabled the installation of apk bot application both from internal resource (Spennymoor Weather) and from file server (Meadowfield Weather). The main finding reveals that it is possible to deliver APK bot application, which has not been tested by any security scans to the mobile device of the victim. What is more Google Play could be employed for this purpose. This seems to be an alarming and obviously a really dangerous behavior, which indicates that the Spennymoor Weather and Meadowfield Weather should never pass through Google Bouncer security scan. Our results also confirmed research findings published in [21] and [23]. Research carried out together with papers listed above also imply that Google Bouncer security tests are solely focused on dynamic application analysis and inspection of AndroidManifest.xml whilst static application analysis is being underestimated. Limitation of the research: current research has been focused on qualitative analyses of Google Bouncer security mechanisms while quantitative analysis has not been performed. Research carried out as a basis of this study has mostly concentrated on techniques allowing bypassing security scans based on dynamic application analysis resulting in the fact that only some aspects of Google Bouncer security mechanisms have been examined. For this reason, it would be beneficial to carry out research of applications published on Google Play focused on automated testing with emphasis on static analyses, which represents at the same time our recommendation for future research. Nevertheless, on the basis of findings published in this article, examination of APK applications published on unofficial sources as file share servers seems to be promising as well.

 ISBN: 978-1-61208-493-0

VIII. DISCLOSURE

During the performed research any data from users has not been collected. All malicious actions were performed only on devices owned by Tomas Bata University in Zlín, Faculty of Applied Informatics. The C&C server and its botmaster interface have been developed by independent offensive security researcher Kamil Vávra (contact: @vavkamil). The botmaster interface has been designed for executing malicious actions based on IP addresses, which ensured that all active targets were devices exclusively owned by Faculty of Applied Informatics. Currently both bot distributional applications published on Google Play are clean, there is no illegitimate part. The Spennymoor Weather and Meadowfield Weather are available and free of charge for everyone. It is our courtesy, how to give warm thanks to users of Google Play.

ACKNOWLEDGMENT

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic within the National Sustainability Programme project No. LO1303 (MSMT-7778/2014) and by the European Regional Development Fund under the project CEBIA-Tech No. CZ.1.05/2.1.00/03.0089 and also by Internal Grant Agency of Tomas Bata University under the project No. IGA/FAI/2016/016.

REFERENCES

[1] Z. Abdullah, M. M. Saudi, and N. B. Anuar, Mobile botnet detection: Proof of concept. s.l., IEEE 5th Control and System Graduate Research Colloquium, 2014.

[2] M. Boodaei, Mobile Malware: Why Fraudsters Are Two Steps Ahead. [Online]. Available at: http://www.trusteer.com/blog/mobile-malware-why-fraudsters-are-two-steps-ahead, 2011.[Accessed 2016 5 10].

[3] B. Choi, et al., Detection of Mobile Botnet Using VPN. s.l., Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 2013.

[4] I. Cornell, How to Create a Complex Password. [Online] Available at: http://www.it.cornell.edu/services/managed_servers/howto/passwords/complexity.cfm, 2012. [Accessed 2016 5 20].

[5] Department of Homeland Security, DHS-FBI Bulletin: Threats to Mobile Devices Using the Android Operating System. [Online]. Available at: https://publicintelligence.net/dhs-fbi-android-threats/, 2013.[Accessed 2016 5 10].

[6] Developers, App Manifest. [Online]. Available at: http://developer.android.com/guide/topics/manifest/manifest-intro.html, 2015. [Accessed 2016 5 9].

[7] Developers, Activity. [Online]. Available at: http://developer.android.com/reference/android/app/Activity.html, 2015. [Accessed 2016 5 15].

[8] Developers, Normal and Dangerous Permissions. [Online]. Available at: http://developer.android.com/guide/topics/security/permissions.html#normal-dangerous, 2015. [Accessed 2016 5 7].

[9] Developers, n.d, Application Fundamentals. [Online]. Available at: http://developer.android.com/guide/components/fundamentals.html, 2015. [Accessed 2016 5 9].

[10] M. R. Faghani, U. T. Nguyen, Socellbot: a New Botnet Design to Infect Smartphones via Online Social Networking. s.l., IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), 2012.

[11] P. Farina, E. Cambiaso, G. Papaleo, and M. Aiello, Mobile Botnets Development: Issues and Solutions. International Journal of Future Computer and Communication, 12, 2014.

[12] G. Geng, et al., The Design of SMS Based Heterogeneous Mobile Botnet. Journal of computers, 1, 2012.

[13] A. Gupta., Learning Pentesting for Android Devices. s.l.:Packt Publishing, 2014.

[14] IDC, 2015, Smartphone OS Market Share, 2015 Q2. [Online]. Available at: http://www.idc.com/prodserv/smartphone-os-market-share.jsp. [Accessed 2016 5 8].

[15] X. Y. Z. S. Jiang, Android Malware. New York, NY: Springer, 2013.

[16] H. Lockheimer, Android and Security. [Online]. Available at: http://googlemobile.blogspot.co.uk/2012/02/android-and-security.html, 2012. [Accessed 2016 5 10].

[17] M. M. Saudi, A new model for worm detection and response : development and evaluation of a new model based on knowledge discovery and data mining techniques to detect and respond to worm infection by integrating incident response, security metrics and apoptosis. Bradford: University of Bradford, 2011.

[18] M. Oulehla, D. Malanik, Techniques Allowing Broadcast Receiver Malware on Android Platform.. Zakynthos, Proceedings of the 19th International Conference on Systems, 2015.

[19] H. Pieterse, M. Olivier, Design of a Hybrid Command and Control Mobile Botnet. The Journal of Information Warfare, 2013.

[20] H. Pieterse, M. Olivier, Android botnets on the rise: Trends and characteristics. s.l., Information Security for South Africa. IEEE, 2012.

[21] S. Poeplau, et al., Execute This! Analyzing Unsafe and Malicious Dynamic Code Loading in Android Applications. s.l., NDSS Symposium, 2014.

[22] A. Polkovnichenko, A. Boxiner, BrainTest – A New Level of Sophistication in Mobile Malware. [Online]. Available at: http://blog.checkpoint.com/2015/09/21/braintest-a-new-level-of-sophistication-in-mobile-malware/, 2015. [Accessed 2016 5 14].

[23] L. Stefanko, Android trojan drops in, despite Google's Bouncer. [Online] Available at: http://www.welivesecurity.com/2015/09/22/android-trojan-drops-in-despite-googles-bouncer/, 2015.[Accessed 2016 5 15].

[24] C. Trout, Android still the dominant mobile OS with 1 billion active users. [Online] Available at: http://www.engadget.com/2014/06/25/google-io-2014-by-the-numbers/, 2015. [Accessed 2016 5 8].

[25] Z. L. W. Wang, C. Wang, How Can Botnets Cause Storms? Understanding the Evolution and Impact of Mobile Botnets. s.l., IEEE INFOCOM 2014 - IEEE Conference on Computer Communications, 2014.

[26] T. Zhao, G. Zhang, and L. Zhang, An Overview of Mobile Devices Security Issues and Countermeasures. s.l.:International Conference on Wireless Communication and Sensor Network, 2014.

 ISBN: 978-1-61208-493-0