Andbot: Towards Advanced Mobile Botnets

Cui Xiang Fang Binxing Yin Lihua Liu Xiaoyi Zang Tianning
Research Center of Information Security
Institute of Computing Technology, Chinese Academy of Sciences
cuixiang@software.ict.ac.cn

***Abstract*—With the rapid development of the computing and Internet access (i.e., using WiFi, GPRS and 3G) capabilities of smartphones, constructing practical mobile botnets has become an underlying trend. In this paper, we introduce the design of a mobile botnet called *Andbot* which exploits a novel command and control (C&C) strategy named *URL Flux*. The proposed Andbot would have desirable features including being stealthy, resilient and low-cost (i.e., low battery power consumption, low traffic consumption and low money cost) which promise to be appealing for botmasters. To prove the efficacy of our design, we implemented the prototype of Andbot in the most popular open source smartphone platform - Android (Google) - and evaluated it. The preliminary experiment results show that the design of Andbot is suitable for smartphones and hard to defend against. We believe that mobile botnets similar to Andbot will break out in the near future, consequently, security defenders should pay more attention to this kind of advanced mobile botnet in the early stage. The goal of our work is to increase the understanding of mobile botnets which will promote the development of more efficient countermeasures. To conclude our paper, we suggest possible defenses against the emerging threat.**

Keywords-Botnet;smartphone;mobile;C&C;URL Flux

I. INTRODUCTION

The term *mobile botnet* refers to a group of compromised smartphones that are remotely controlled by *botmasters* via *C&C* channels. While PC-based botnets, as common platforms for many Internet attacks, have become one of the most serious threats to Internet, mobile botnets targeted for smartphones are not as popular as their counterparts for a variety of reasons including resource issues, limited battery power, and Internet access constraints, etc. Consequently, both the occurrence of practical mobile botnets and corresponding research on them are very limited. However, this could change with the recent surge in popularity and use of smartphones. Smartphones are now widely used by billions of end users due to their enhanced computing ability and efficient Internet access. Moreover, smartphones always store a large amount of sensitive personal data and are often used in online payment. The emergence of open-source smartphone platforms such as Android and third-party applications made available to the public also provides more opportunities for malware creators. Therefore, smartphones have become one of the most attractive targets for hackers. Since the appearance of Cabir, the first mobile worm (which was introduced in 2004), we have witnessed a significant evolution in mobile malware. Although the number of mobile malware has been growing steadily, their functionalities have remained simple until the development of the first mobile botnet in 2009. The mobile botnet, SymbOS.Yxes [1], targets Symbian and exploits a simple HTTP-based C&C. Later the same year, Ikee.B [2], which targets jailbroken iPhones and has a C&C mechanism similar to SymbOS.Yxes, was released. In December 2010, the first Android botnet, Geinimi, broke out mainly in China, still using similar HTTP-based C&C. Although advanced mobile botnets have not been witnessed in the main population of smartphones, we believe it is just a matter of time. Mobile botnets are presently posing serious threats for both end users and cellular networks [7]. Consequently, investigations into how mobile botnets work, as well as how they may be developed and stopped, represents an important area of research.

A. The Challenges of Constructing a Mobile Botnet

There are several differences between smartphones and PCs. These differences lead to a number of challenges in the construction of a mobile botnet [3, 7]. (1). the battery power is rather limited on smartphones when compared with PCs. If the battery power consumption speed exceeds user expectations, the battery exhaustion is likely to be noticed by the user, leaving the bot open to detection. (2). the cost of smartphones is an extremely sensitive area for many users. If data costs begin to exceed the amount that the user had expected or agreed to pay, the bot could also be detected. (3). if C&C consumes an abnormal amount of network traffic, the abnormity is likely to be noticed. (4). the absence of public IP addresses and a constant change in network connectivity makes the robust P2P-based C&C in PC-based botnets impractical, and potentially impossible, in smartphones.

B. The Proposed Andbot

Considering the above challenges faced by botmasters, the design of a practical mobile botnet, from our understanding, should consider the following questions: (1). How to design a stealthy C&C channel to make detection more difficult? (2). How to recover the C&C channel in case all critical resources are destroyed (i.e., DNS redirected, rendezvous servers shutdown by defenders)? (3). How can noticeable factors such as monthly charges, traffic, and battery power consumption be decreased to an acceptable degree to prevent detection by infected users? (4). How to prevent (or make it harder) the botnet away from hijacking even if the bot is completely reverse analyzed and all the critical resources are controlled by coordinated defenders.

By considering all the challenges listed above, in this paper, we present our research on the possible design of an advanced mobile botnet named *Andbot* on smartphones running the Android operation system. The proposed Andbot has the following features:

- **Stealthy:** Using HTTP-based URL Flux protocol, it will only access Internet in background.
- **Resilient:** (1). Resistant to most of public known defense strategies such as DNS sinkhole, malicious

commands injection, IP blacklist and C&C server shutdown, etc; (2). Recover C&C in an accepted time delay in the case that crucial resources are temporally unavailable.

- **Low-Cost:** Low money costs, low traffic and battery power consumption.
- **Commands supported:** CallHome, SMS Phishing and Filtering, DDoS, Information Theft, Sleep.

C. Paper Organization

The rest of the paper is organized as follows. Section II introduces related studies. Section III introduces the architecture of Andbot. Section IV discusses the C&C design. In Section V, we study the effectiveness and efficiency of Andbot. We present possible defenses against Andbot in Section VI. We give a few future plans and conclude the paper in Section VII.

II. Related Work

Botnets have been an active research topic in recent years. Current research on botnets is focused primarily on detection, measurement, tracking, mitigation, and future botnet prediction. Our research belongs to the last category.

Wang et al. [11] presented the design of an advanced hybrid peer-to-peer botnet. Vogt et al. [12] presented a "super-botnet" - that works by inter-connecting many small botnets together in a peer-to-peer fashion. Ralf Hund et al. [13] introduced the design of an advanced bot called Rambot, developed from the weaknesses they found when tracking a diverse set of botnets. Starnberger et al. [14] presented Overbot, which uses an existing P2P protocol, Kademlia, to provide a stealth C&C channel. Singh et al. [15] evaluated the feasibility of exploiting email communication for botnet C&C.

Nevertheless, few research works have studied how botmasters might design their advanced C&C for smartphones-based botnets. Singh et al. [5] evaluated the feasibility of using Bluetooth as a medium for botnet C&C. We believe that this approach could be effective only when the mobile botnet is extremely huge (i.e., more than ten million), therefore, we focus our research on Internet based C&C. Mulliner et al. [3] proposed a SMS-HTTP hybrid C&C. The main idea of the hybrid schema is to split the communication into a HTTP and a SMS part. The encrypted and signed commands file is uploaded to a website and the corresponding URL is distributed via SMS. Zeng et al. [4] utilizes a SMS-based C&C with a P2P topology. Our work is complimentary to these approaches in that: (1). The SMS-based C&C (especially P2P topology) will inescapably cause excessive fees to users which leads to detection; and (2). The simple HTTP-based C&C scheme suffers a single-point-failure. As such, our study compliments this existing research, as we have eliminated the single-point-failure problem to some degree thanks to URL Flux.

III. Overview Of Andbot

A. The Command and Control Architecture of Andbot

Andbot uses a centralized C&C topology, that is, it connects to a fixed number of C&C servers and obtains commands from them. Compared to traditional IRC- and HTTP-based centralized botnets (see Fig. 1), it is easy to see that Andbot C&C shown in Fig. 2 adds an abundant mechanism – if one username(black cycle) is blocked or fails to register it in Microblog, other usernames(white cycles) could be registered instead, making the C&C more resilient.

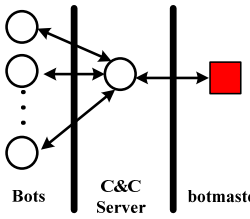

Fig. 1 IRC/HTTP-based C&C

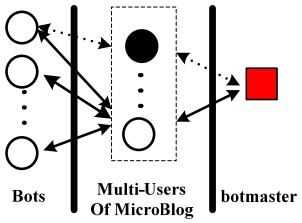

Fig. 2 URL Flux-based C&C

To obtain the capability described above, Andbot hard-codes a public key, a number of Web 2.0 addresses (i.e., domain names), and a Username Generation Algorithm (UGA). In order to find commands, Andbot first connects to one of the Web 2.0 servers and then tries to visit the users generated by UGA, one by one. If the visited user exists, the most recent messages will be verified using the hard-coded public key. If passed, the messages are convinced to be issued by the authorized botmasters. We have named this kind of C&C technology "*URL Flux*" (see Fig. 2 and Fig. 3), based on the name convention of Domain Flux (see Fig. 4) —a powerful C&C technology used by the Conficker [20] botnet which targets the Windows operating system. In a Domain Flux scheme, bots must hard-code a public key and a Domain Generation Algorithm (DGA). Bots try to connect to and download a command file from the generated domains one by one, and then authenticate the downloaded command file using the hard-coded public key.

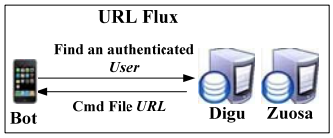

Fig. 3 URL Flux

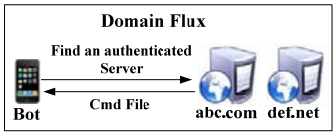

Fig. 4 Domain Flux

The advantage of URL Flux is that it doesn't require publicly available servers and the corresponding domain name to be bought. Moreover, the C&C servers used are always fairly robust and easier to bypass Firewalls.

B. Low Cost

A significant rise of the phone bill, traffic, or excessive consumption of battery power will lead to investigation of the cause, and thus may lead to bot detection [3, 8-10]. Thus, several methods were designed to minimize the consumption of the above resources: IP-only C&C, RSS and GZIP compression, URL Caching, and *Sleep* command.

- **IP-only C&C:** Each step of C&C depends on TCP/IP communication other than SMS or Bluetooth. Obviously, Internet access is an essential requirement for Andbot. Luckily, most of current smartphones are easy to access Internet.
- **RSS and GZIP compression:** Many Microblogs support browsing messages (such as "tweet" in Twitter) which use RSS and respond using GZIP compression. These capabilities fall squarely within the low traffic consumption requirement of mobile botnets. Our experiments in Section V prove that these methods could reduce traffic consumption significantly.
- **URL Caching:** Once one authorized URL, which points to correctly signed commands, is found, Andbot will cache it in its period of validity.
- **Sleep:** If a botmaster decides to refrain from publishing new commands for some time, or to minimize the interval between two commands, a *Sleep* command can order the bots to sleep so as to decrease the consumption of resources in smartphones.

C. Supported Commands

Andbot implements necessary commands which are different from those of PC counterparts. It is due to these capabilities that mobile botnets pose a more dangerous threat than PC botnets. The implemented commands and associated descriptions are listed in Table. 1.

Commands	Description
.CallHome#Channel#Address	Call Home to "Address" via "Channel" (i.e., HTTP, Email, and SMS).
.SMSDoS#MobileNumber#Num #Random#Content#Len	DoS "MobileNumber" by sending Num SMS ether using fixed "Content" or generating "Random" content no longer than "Len".
.SMSSpread#Content#Dest	Sending SMS with "Content" to either all contacts in address book or special user based on "Dest". The "Content" is usually a phishing message containing a valid URL.
.MonitorSMS#MobileNumber# Num#Channel#Address	Monitoring new coming SMS, recent SMS or special SMS based on "MobileNumber" and send the SMS to botmaster via HTTP, Email, or SMS determined by "Channel" and "Address".
.GenSMS#FakeFromNumber# DateTime	Generating a fake SMS from "FakeFromNumber" on "DateTime", this is useful for Phishing.
.DenySMS#FromNumber	Intercpting any SMS from "FromNumber". This command is used to filter some warning SMSs from special ISP number (i.e., +8610086).
.RelayCmd#CipherCmd#Num# MobileNumberList	Relay the "CipherCmd" to "Num" bots in "MobileNumberList". The "CipherCmd" includes both ExpireDate and Cmd.
.Sleep#Seconds	Let the bots sleep for some time.

Table. 1 The Supported Commands of Andbot

IV. Botnet Command And Controle Design

C&C is the most important part of a botnet. Although many advanced C&C strategies designed for PC botnets have been deployed by many successful botnets (such as the Storm, Conficker, Waledac, and Stuxnet), we believe that none will succeed as mobile botnets when faced with the limitation factors associated with smartphones.

Considering the problems with deploying mobile botnets, we design the C&C of Andbot sensitively and thoroughly. For example, to gain the element of stealth, Andbot connects to Internet only when the smartphones in sleeping state; to gain resiliency, Andbot uses URL Flux rather than a simple HTTP-based C&C; to minimize the consumption of network traffic, Andbot exploits RSS and GZIP compression as opposed to visiting Web 2.0 websites directly; to avoid battery exhaustion attack [8, 9, 10], Andbot does not start C&C communication frequently - especially when receiving a *Sleep* command.

Like many other botnets [11, 13], Andbot also uses RSA to authenticate commands, so botnet hijacking is not a major problem. Since the public-key based authentication has been deployed by many current C&C botnets, we won't explain it in detail.

A. Command and Control Design

To outline the rough sketch of the complete C&C procedures of Andbot, the following list provides the sequence of operations for both botmaster and Andbot in Fig. 5.

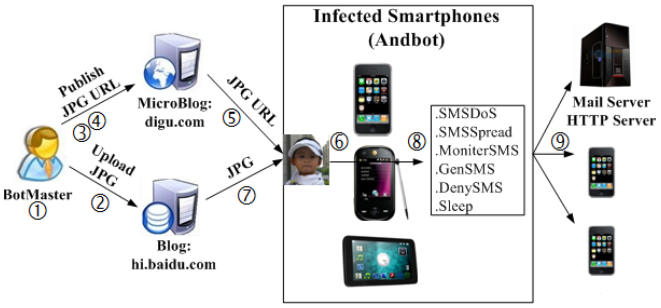

Fig. 5 The C&C Architecture of Andbot

① Botmaster encrypts and signs the commands to be issued, then binds the ciphertext with a small JPG file.

② Botmaster uploads the JPG file to a Blog, then compresses the URL, denoting the shorten URL as J1.

③ Botmaster combines "StartDate", "ExpireDate" and J1 together, then encrypts, signs, and encodes it.

④ Botmaster publishes the ciphertext to the homepage of a Microblog user, which is registered in advance, and denoted as U2.

⑤ Andbot visits users on Microblog one by one using UGA until finding U2.

⑥ Andbot decrypts out J1 based on recent "tweets" of U2.

⑦ Andbot downloads JPG based on J1.

⑧ Andbot recovers plaintext commands from the JPG.

⑨ Executes commands.

- **Prepare a JPG File Containing Encrypted and Signed Commands:** In the first step, a botmaster encrypts the commands to be published using RC4 then signs it using her private key which is corresponding to the public key hard-coded in Andbot. Second, the botmaster appends the ciphertext and its length to the end of a JPG file (there is no need to use watermark because the JPG file is very small and recovering the hided information costs complex code).

  Upload the JPG file and compress the URL: After generating the special JPG file, the botmaster uploads it to a public website, such as a blog or picture-hosting site. Considering the maxlength limit in Microblog, the botmaster may compress the raw URL using popular services such as bit.ly and tinyurl.com. In this way, the length of the shortened URL will be more easily controlled.

- **Prepare Microblog Message Containing The JPG URL and Time Constraint:** The main role of Microblog is to publish the URL of the JPG file containing the encrypted and signed commands. Moreover, in order to countermeasure the *replay attack*, a *StartDate* and *ExpireDate* are indispensible. Only when the current date is later than *StartDate* and earlier than *ExpireDate* the URL is considered to be valid and active. We explain the interface among botmaster, Andbot, Microblog, and blog in Table. 2.

Interface		**Description**
StartDate		The Cmd will become valid from StartDate
ExpireDate		The Cmd will expire after ExpireDate
JPG URL	Original	The URL of the uploaded JPG File (i.e., http://hiphotos.baidu.com/peakxp/pic/item/b157b21edc0ffc4f4334176c.jpg)
	Shorten	The shorten URL using bit.ly (i.e., http://bit.ly/esNWwF)
Input		StartDate#ExpireDate# Shorten JPG URL (# denote conjunction here, . i.e., 2011011020110201http://bit.ly/esNWwF)
Output		Base64 (Sign (Hash (Input))#Encrypt (Input)). i.e. , part1=NJ0pU60znHjo5KfDcKS8Rv5OFoPdpfRvTWr59a049pwFC6xpdhu1YZCJ9/UhpBkKK1DSsYKCa2OZ2VYGeoy8S1rm+x04JguhbjXAlH8LXpc45jl1GJ part2=JW2VLj6bvx6wkRWPYpb2iAymlpvEgXWUs5e5zAfUPVVluG+QYUmMte/wjjXQv+WVH80EOTs6ISePgUYq/pI7EY2vKfeTDqr0BQZKO9uxo=
Microblog Message		The first Microblog Message is part1 and the next is part2

Table. 2 The Interface Description with Examples

- **URL Flux-based C&C:** Andbot first connects to several pre-defined Microblogs, and then requests the RSS of special users. If the response is verified successfully using the hard-coded public key and the messages are not expired, Andbot will download the JPG file using the decrypted URL. Again, Andbot will verify the JPG file using the hard-coded public key and decrypted the message using its RC4 symmetric key. Finally, Andbot obtains and executes the plaintext commands. The UGA and URL Flux Algorithm are shown in Fig. 6 and parts of the result of UGA are shown in Fig.7. There are a number of usernames that could be generated on a monthly or half-yearly basis. URL Flux uses this mechanism to enhance the counterattack capability.

B. *Advantage Analysis*

- **Stealth:** Since all the C&C traffic uses HTTP, which is the most popular Internet traffic, it can be considered rather stealthy in its ability to bypass firewalls. In addition, Andbot only accesses background Internet, bypassing the warning of traffic monitoring softwares.

  Resilience: The C&C servers are high-performance websites which could serve millions of communications concurrently. Also, the response from Microblog will change given that the "*StartDate*" and "*ExpireDate*" are changing, making signature-based detection difficult. In case the username generated by UGA is blocked by Microblog providers, a botmaster could generate and use another username in the same or different Microblog. Furthermore, because the private key is owned only by botmasters, either injecting malicious commands or replay expired commands is impossible.

- **Low Cost:** The following factors all contribute to the minimal cost impacts of Andbot: (1). Considering the fact that WIFI is often provided free of charge (while GPRS is expensive), Andbot will preferably select WiFi. Luckily enough in this example, Android will select WiFi automatically if both WiFi and GPRS are available. (2). Andbot will preferably select RSS of Microblog to retrieve the content because RSS produces less traffic. (3). Andbot requests GZIP format to shrink the size of response packets. (4). Andbot uses a URL caching mechanism to save the successfully verified URL until the current time reaches "*ExpireDate*". Thus, one successful URL Flux addressing result could theoretically be used at the discretion of the botmaster. (5). Botmasters could publish a *Sleep* command. (6). In SMS-based C&C, the bots will inescapably cost a lot of money when sending a large amount of SMS. In our work, we only use GPRS and WiFi occasionally; thus, the money cost could be controlled in an acceptable scope.

```
UGA(UserName)
1. current _ date = GetCurrentDate()
2. UserName = Gen _ User _ Name(current _ date)

URLFlux()
1. For MicroBlog 1 to N
2.   If connect(MicroBlog) == Success
3.    While MaxUserNum-- > 0
3.       UGA(UserName)
4.       If DecodeAndVerify(in UserName, out URL)==Success
5.          DownloadJPG(in URL, out JPGFile)
6.          DecryptAndVerify(in JPG, out PlainCmd)
7.       Else
8.          Continue
```

Fig. 6 The UGA and URL Flux Algorithms

Validate in a half year:

http://digu.com/statuses/rss/pbipnv132545.rss

http://digu.com/statuses/rss/tk1074939514.rss

http://digu.com/statuses/rss/dfhgp3782858.rss

...

Validate in a whole month:

http://digu.com/statuses/rss/zkng61647311.rss

http://digu.com/statuses/rss/d11242809261.rss

http://digu.com/statuses/rss/rfit14943355.rss

http://digu.com/statuses/rss/lebzpztb2139.rss

http://digu.com/statuses/rss/xbj131123422.rss

...

Fig. 7 The example output of UGA

V. EVALUATION OF ANDBOT

To evaluate the functions and performance of Andbot, we've conducted preliminary experiments, the results show that Andbot can work properly and has desirable features. The experiments were conducted in a controlled environment, and the propagation function was carefully processed to ensure the bot would be unable to spread to other mobile devices outside of those used for the experiment.

A. Experiment Environments

- Four Android-based smartphones were tested: HTC Legend, Motorola xt502, Motorola xt702 and Samsung i5700.
- Web 2.0 Services: register some Microblog and blog accounts.
- Web Server: a WAMP 2.0 was deployed to provide MySQL database and Web service.
- Email: Register one email to recycle SMS.

Among the four types of smartphones, the first one uses Android 2.2 while the others use Android 2.1. Web 2.0 services which are core resources of C&C. Web server and Email are just used to receive the "stolen" information, such as IMEI, IMSI, OS, Version and SMS, from the "infected" smartphones.

B. Functionality Test

- **Autorun and Bypassing Security Softwares:** (1). To be more practical, we hide Andbot inside a popular game named *MixedColor*. Starting the malicious game manually, it runs correctly and friendly, and Andbot is activated and executes in the background; (2). After rebooting Android, the malicious game can auto-start with activated Andbot functions; (3). We repeated the above operations when installing three kinds of mobile AVs including f-secure, Netkin and 360 safeguard in turn. No warnings appeared, meaning Andbot survived successfully.
- **The Correctness of supported Commands:** (1). We carefully processed the address book and SMS inbox in our test smartphones; (2). We combined one or more commands from the supported commands; (3). We issued the above commands to verify their correctness. Although all the commands were executed successfully, many questions raised during the experiments which have proven helpful in revisions of our design. For example, should Andbot only remain active when in sleeping state? The answer is, yes, because some traffic monitor software will show the current network activity. Should Andbot automatically switch to ASN (i.e., from CMWAP to CMNET)? The answer is yes, as well, because CMWAP could not access Internet, and Andbot must switch the network option by itself.

C. The C&C Cost Evaluation

The traffic cost during C&C is the most important evaluation factor. First, we define the necessary parameters (see Table. 3) which must be considered in C&C.

Parameters	Description
α	The interval between two commands requesting
β	The half-year username count
γ	The month username count
ω	The total num of different Microblogs
θ	The flag that indicates if RSS and GZIP should be used. if θ=1, they will be used, otherwise 0
δ	The flag that indicates if bot should keep active only when sleeping, if δ=1, bot will keep active, otherwise 0

Table. 3 The Parameters of C&C

In the experiment, we assigned the parameters α=10mins, β=10, γ=50, ω=2(digu.com and zuosa.com), θ=1, δ=1 and use "hi.baidu.com" (a famous blog in China) to host JPG files. To explain simply and clearly, we only show the real C&C cost results about Digu (see Table. 4). The result of Zuosa is a little slower than Digu. We can see if a user exists (i.e., for username *tk1074939514)*, the average time from first packet to last packet spends only *2.706* seconds averagely. Andbot needs to send a packet with *164* bytes payload to Microblog server and get a response *with 1062* bytes payload. Considering the TCP connection and disconnection traffic, and all the packet headers, the total traffic is *1902* bytes. Remember, when Andbot successfully finds a URL, it will cache it for future use until reaching ExpireDate. So the total traffic is fairly low even after 12 hours.

In general, β and γ are big while a botmaster could only register a few user accounts, so Andbot will inescapably visit a

large amount of non-exist users. Therefore, we need to evaluate the performance when only part of users exists (i.e., 5%, 10% and 50% half-year username exists). The results of time delay and traffic cost are shown in Table.5. When a few usernames exist, Andbot only consumes several thousand of bytes to find the correctly signed JPG URL. Then, Andbot begins to download the JPG file (see Table.6).

From the preliminary results, we can see that Andbot consumed a very small amount of resources, making detection difficult because obvious abnormalities may not be seen by end users.

SubURL	User Name	Gzip	Avg. Time Delay(s)	Request/Response/Total Traffic(Byte)
/statuses/rss/pbipnv132545.rss	Not Exist	No	7.618	164/348/1188
/statuses/rss/tk1074939514.rss	Exist	No	13.745	141/1972/2995
/statuses/rss/tk1074939514.rss	Exist	Yes	2.706	164/1062/1902

Table. 4 The Performance Evaluation for one-visiting

C&C Type	Available Username Num		Time Delay(S)	Total Traffic(KB)
Locate the first authorized JPG URL	Half Year	5%	30.61s	12.87
		10%	14.85s	7.07
		50%	4.46s	2.43

Table. 5 The Performance Evaluation for multi-visitings

C&C Type	JPG File Size(Byte)	Cipher Cmd Len(Byte)	Time Delay(S)	Traffic Cost(Byte)
Download JPG File	2326	213	3.06s	3705

Table. 6 The Performance for Downloading the JPG File

VI. Defense Against Andbot

We introduce possible defense in three ways. First, an internationally coordinated cooperation channel should be set up quickly to identify and defend against this technology; second, we should pay more attention to the management of software publications; third, we should infiltrate mobile botnets to monitor their activities in time.

- **Building International Coordinated Mechanism:** The C&C of Andbot relies on Web 2.0 Services. For this reason, defenders should focus their defense effort on publicly available Web 2.0 services such as Microblog, blog, Google App Engine, etc. This effort can prevent these services from being abused. In the case that abnormalities are detected, there should be a coordinated channel such as CERT to stop the corresponding services.

- **Monitoring at SMSC side and Verify in Cloud Sandboxes/VMs:** In general, current mobile malware mainly spread via social engineering such as sending phishing SMS, or publishing malicious softwares on websites. In the first case, defenders may deploy a worm detection system at SMSC level using a similar algorithm to Autograph [16], Early-birds [17], etc. After obtaining the suspicious URL embedded in SMS, defenders could download the softwares and verify them inside Cloud Sandboxes or Virtual Machines (VMs). If the softwares are found to be malicious, the signature should be generated automatically, allowing defenders to take some countermeasures. In the second case, defenders should pay more attention to software distribution management, either using Cloud Sandboxes/VMs or using several updated AVs to verify new softwares before they become available to mobile users.

- **Infiltration:** Since most bots on smartphones must find commands in an active way, all of them are inescapably vulnerable to an infiltrator [18]. After a defender's analysis of the C&C of Andbot, an infiltrator can be written using the same URL Flux protocol. In this way, defenders are able to track the botnet activities.

VII. Future Work and Conclusions

From the defense discussion in previous section, we see that Andbot still has some shortcomings. We plan to make several improvements for Andbot as below.

- **Dynamic UGA:** In Andbot, a static UGA is used which can satisfy most cases. However, imagining an extreme case, defenders can register or block all the usernames which could possibly be generated by Andbot, whether now or in the future. With this defense, Andbot will lose control. To prevent this type of attack, a Dynamic UGA (DUGA) should be developed. There are many technologies suitable for DUGA. For example, a DUGA could first query the most active and popular topics from Google, Twitter, etc., and then use the retrieved keywords as the seed of UGA.

- **Time-Space Deviation:** BotMiner [19] detects botnets using the time-space similarity. This is because they believe bots always work in a coordinated way, which leads to time and space similarities in communication content and patterns. Therefore, to avoid this kind of detection, Andbot needs to randomize its C&C communication contents to eliminate space similarity (i.e., injecting packet and flow-level noise), and add a random delay to eliminate time similarity when responding to some interactive commands.

- **Emergency C&C:** Andbot has a cache and sleep mechanism which helps it to minimize resource consumption. Nevertheless, some tasks may be urgent for Andbot to perform. As such, there should be an emergency SMS-based C&C channel to issue urgent commands rapidly in a PUSH style. Obviously, the SMS-based C&C should not be used except in emergency situations.

As smartphones continue to gain more capabilities, they become attractive targets to hackers. To be well prepared for the promise attack, we, as defenders, should study mobile botnets attacking techniques that are likely to be developed by botmasters in the near future. In this paper, we presented the design of a *stealthy*, *resilient*, and *low cost* mobile botnet called Andbot, and evaluated its efficacy thoroughly. Our preliminary results show that the proposed Andbot is feasible and effective. To defend against such a mobile botnet, we suggest several possible countermeasures. In the future, we will invest more research on how to fight against this kind of advanced mobile botnet.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their helpful comments for improving this paper. This work is partly supported by the National High Technology Research and Development Program (863 Program) of China under grant (No. 2007AA010501) and the National Natural Science Foundation of China under grant (No. 61070186 and No. 61070026).

REFERENCES

[1] Axelle Apvrille. Symbian worm Yxes Towards mobile botnets. http://www.fortiguard.com/papers/EICAR2010_Symbian-Yxes_Towards-Mobile-Botnets.pdf

[2] P.A. Porras, H. Saidi, V. Yegneswaran, "An Analysis of the iKee.B iPhone Botnet," in Proceedings of the 2nd International ICST Conference on Security and Privacy on Mobile Information and Communications Systems (Mobisec), May 2010

[3] C. Mulliner, J.P. SeifertIn. Rise of the iBots: 0wning a telco network. In the Proceedings of the 5th IEEE International Conference on Malicious and Unwanted Software (Malware) Nancy, France 19-20 October, 2010

[4] Yuanyuan Zeng, Xin Hu, Kang G. Shin, "Design of SMS Commanded-and-Controlled and P2P-Structured Mobile Botnet", *The University of Michigan, Ann Arbor, MI 48109-2121, U.S.A. 2009.*

[5] K. Singh, S. Sangal, N. Jain, P. Traynor and W.Lee, "Evaluating Bluetooth as a Medium for Botnet Command and Control," in Proceedings of the International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA), July 2010.

[6] C. Mulliner, "Fuzzing the Phone in your Phone," http://www.mulliner.org/security/sms/feed/smsfuzz_26c3.pdf

[7] P. Traynor,M. Lin, M. Ongtang, V. Rao, T. Jaeger, T. La Porta and P. McDaniel, "On Cellular Botnets: Measuring the Impact of Malicious Devices on a Cellular Network Core," in ACM Conference on Computer and Communications Security (CCS), November 2009.

[8] D. C. Nash, T. L. Martin, D. S. Ha, and M. S. Hsiao, 3 Towards an intrusion detection system for battery exhaustion attacks on mobile computing devices, 4 in PERCOMW '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops. Washington, DC, USA: IEEE Computer Society, 2005, pp. 141+145.

[9] Hahnsang Kim, Joshua Smith, and Kang G. Shin. Detecting energygreedy anomalies and mobile malware variants. In MobiSys, 2008.

[10] Davis, N. Battery-based intrusion detection. In: Proceedings of the Global Telecommunications Conference (2004) A.-D.

[11] P. Wang, S. Sparks et al. An advanced hybrid peer to peer botnet. Proc. of the HotBots'07, First Workshop on Hot Topics in Understanding Botnets, Cambridge, MA, 2007.

[12] R. Vogt, J. Aycock, and M. Jacobson, "Army of botnets," Proc. of 14th Annual Network and Distributed System Security Symposium (NDSS'07), 2007.

[13] R. Hund, M. Hamann and T. Holz. Towards Next-Generation Botnets. Proc. of the fourth European Conference on Computer Network Defense (EC2ND 08), 2008.

[14] G. Starnberger, C. Kruegel, and E. Kirda, "Overbot - a botnet protocol based on kademlia," in Proc. of the 4th Int. Conf. on Security and Privacy in Communication Networks (SecureComm 08). September 2008.

[15] Kapil Singh, Abhinav Srivastava et al. Evaluating Email's Feasibility for Botnet Command and Control // Proceedings of The 38th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2008). Anchorage, Alaska. June 2008.

[16] H.-A. Kim and B. Karp. Autograph: toward automated, distributed worm signature detection. In Proceedings of the 13th USENIX Security Symposium, August 2004.

[17] S. Singh, C. Estan, G. Varghese, and S. Savage. Automated worm fingerprinting. In Proceedings of the 6th ACM/USENIX Symposium on Operating System Design and Implementation (OSDI), Dec. 2004.

[18] Juan Caballero et al. Dispatcher: Enabling active botnet infiltration using automatic protocol reverse-engineering. In ACM CCS, 2009.

[19] Gu G, Perdisct R, Zhang J and Lee W. BotMiner: clustering analysis of network traffic for protocol- and structure- independent botnet detection // Proceedings of the 17th USENIX Security Symposium (Security'08). San Jose, CA, 2008:139-154

[20] P. Porras, H. Saidi, and V. Yegneswaran. A Foray into Conficker's Logic and Rendezvous Points. In USENIX Workshop on Large-Scale Exploits and Emergent Threats,2009.