This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings

Efficient Detection of Bots in Subscribers' Computers

José Brustoloni, Nicholas Farnan, Ricardo Villamarín-Salomón and David Kyle
Dept. of Computer Science, University of Pittsburgh
210 S. Bouquet St. #6111, Pittsburgh PA 15260 – USA
{jcb,nlf4,rvillsal,dkyle}@cs.pitt.edu

***Abstract*— We investigate how an ISP can efficiently detect bots in its subscribers' computers, possibly as a value-added service or to prevent collateral damage to its infrastructure. By causing an ISP's email servers and network links to get clogged or blacklisted, bots reduce the quality of service the ISP provides to its subscribers. We describe DNS Flagger, a novel device for ISP bot detection, and evaluate its efficiency. DNS Flagger matches subscribers' DNS traffic against IP and DNS signatures. In real-time experiments, we found that, on average, major anti-virus programs (AVs) detected only 59% of freshly caught bots, while DNS Flagger detected 73.1% or 91% of those bots, respectively on hosts that do not or do also have a major AV. There were no false alarms. Because its processing involves only a small fraction of all network traffic and can be performed at very high speed, a single DNS Flagger can handle hundreds of thousands of subscribers.**

I. Introduction

Bots are forms of malware that propagate like viruses or worms and receive commands from a botmaster after infection [1], [7]. Bots usually obtain commands by contacting a botnet command and control (C&C) server. Bots are a large and rapidly growing threat. It has been estimated that 7% of the hosts connected to the Internet are infected by a bot [6]. Botmasters use bots for a variety of attacks. Attacks against infected hosts often hurt their performance and may include capturing private information or credentials for identity theft. Attacks against remote victims may include spam, distributed denial-of-service (DDoS) attacks, phishing, and click fraud. These attacks also frequently cause collateral damage to Internet service providers (ISPs). By causing an ISP's email servers and network links to get clogged or blacklisted, bots reduce the quality of service the ISP provides to its subscribers.

Home computers are specially vulnerable to bot infection. Home computers often are not regularly maintained and may lack security patches, anti-virus software (AV), or up-to-date virus signatures. Firewalls and NAT routers are common in homes, but they typically are ineffective against infection via malicious email attachments, file downloads, or browser vulnerabilities.

We investigate how an ISP can efficiently detect bots in its subscribers' computers, possibly as a value-added service or to avoid collateral damage from bots. Efficiency in this case involves trade-offs between detection rate, false alarm rate, and scalability. In order to provide value to subscribers, ISP bot detection and false alarm rates should be at least as good as those of major AVs, and significantly better for hosts that already have a major AV. ISP bot detection must be able to achieve such detection and false alarm rates in real-time while considering the network traffic of hundreds of thousands of subscribers.

This paper describes DNS Flagger, a novel device for ISP bot detection, and evaluates its efficiency. DNS Flagger matches subscribers' DNS traffic against IP and DNS signatures with, respectively, the IP addresses and domain names of blacklisted C&C servers. Because such processing involves only a small fraction of all traffic and can be performed at very high speed, a single DNS Flagger can handle an ISP's entire metropolitan area. We contribute DNS signature harvesting, a low-cost technique for obtaining DNS signatures from known IP signatures (which often are more readily available), and show that it signficantly improves bot detection. In real-time experiments, we found that, on average, major AVs detected only 59% of freshly caught bots, while DNS Flagger detected 73.1% or 91% of those bots, on hosts that do not or do also have a major AV, respectively. There were no false alarms.

This paper is organized as follows. Section II summarizes how bots can be caught and analyzed. Section III describes local DNS signature harvesting. Section IV presents DNS Flagger's design and implementation. Section V describes our methodology for measuring bot detection rate in real time. Section VI reports our experimental results. Section VII discusses related work, and Section VIII concludes.

II. Bot capture and analysis

This section provides background information on how bots can be captured and analyzed.

Bots that propagate like worms can be caught in honeypots. Honeypots are servers that have no legitimate use and are deliberately configured with vulnerabilities that enable infection [2]. Some malware propagates to a vulnerable client (e.g., browser) when the latter visits an infected server. Such bots may be collected by so-called client honeypots [22]. Bots that propagate like viruses can often be captured by spamtraps. Spamtraps are email accounts that are advertised on the Web but have no legitimate use. Nontext attachments in email messages received in such accounts often are malware.

Once a malware specimen is captured, it can often be identified by an AV. AV detection rates depend on how up-to-date the AV's code signatures are. Even with the latest signatures, however, AVs typically fail to detect many freshly

978-1-4244-3435-0/09/$25.00 ©2009 IEEE

caught specimens. Submitting a same specimen to multiple AVs significantly increases the likelihood of identification. VirusTotal [21] provides Web and email interfaces for submitting specimens in parallel to 32 different continuously updated AVs. AV vendors typically obtain new code signatures by disassembling and analyzing submitted specimens that are not identified by existing signatures.

IP and DNS signatures, on the other hand, may be obtained by running specimens in a sandbox [20]. A sandbox contains one or more hosts on which malware can run. Sandbox hosts connect to the Internet via a special gateway that permits and records communication while limiting attacks toward the Internet. Analysis of communication emanating from a sandbox may reveal addresses and possibly names of C&C servers.

Security vendors typically commercialize malware signatures by subscription. The cost of updates may increase with frequency (e.g., hourly instead of daily). There are also several noncommercial alliances of system administrators who collaborate on collecting and analyzing malware. For example, MWCollect is an alliance with more than 30 honeypots distributed throughout the world for collecting malware [13]. Another example is Shadowserver, an international alliance of volunteers who collect, analyze, and share intelligence on botnets and other threats [18].

III. Obtaining and maintaining DNS signatures

A botmaster may register one or more domain names for C&C servers. Each name may in turn be mapped to many IP addresses, one for each instance of the C&C server. Ideally, a bot detector would use a blacklist containing all C&C domain names. In practice, however, only partial information may be available. This section discusses heuristics for gleaning C&C intelligence from partial clues.

Local DNS signature harvesting consists in (1) positioning a bot detector such that it can observe DNS traffic between (a) monitored DNS clients and (b) DNS resolvers and servers, and (2) including in a blacklist new names that, according to replies to DNS address queries, resolve to addresses already in the blacklist. This heuristic may improve detection rate, especially in two cases. First, if initially only IP signatures of C&C servers are known (e.g., in the form of Snort or firewall rules). Second, if the botmaster aliases a same C&C server whose IP address is known to multiple domain names, not all of which may initially be known. In both cases, the heuristic records previously unknown names that later resolve to previously unknown addresses, enabling C&C detection that otherwise would not be possible.

To prevent spoofed DNS replies from polluting the blacklist, DNS replies should be statefully filtered [15] (in particular, the reply's identifier and destination should match the identifier and source of a recent query). Note also that reverse DNS lookups and the reverse heuristics are insecure. An attacker can use a DNS PTR record to associate his blacklisted address (eg., 12.34.56.78) to the name of a legitimate site, e.g., www.google.com. Alternatively, an attacker can use a DNS A record to point a blacklisted domain name (e.g., www.attacker.com), which he controls, to the address of a legitimate site, e.g. 74.125.39.99 (which actually corresponds to www.google.com). To avoid false alarms, local DNS signature harvesting includes in the blacklist only new *names* corresponding to addresses already in the blacklist, received in replies to DNS address queries. It does *not* update the blacklist based on new names in PTR records or new addresses in A records, which attackers can easily spoof.

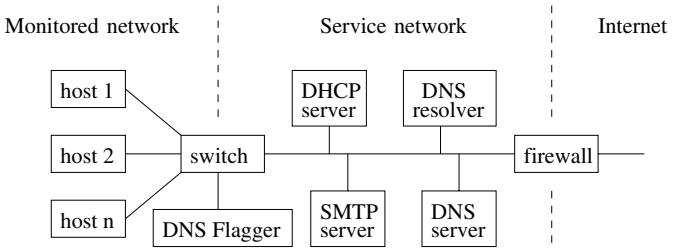

Fig. 1. DNS Flagger architecture

Local DNS signature harvesting can be considered a specialized, low-cost form of passive DNS replication [23]. The latter captures DNS replies and stores DNS records therein contained in databases independently replicating authoritative DNS servers. Unlike local DNS signature harvesting, passive DNS replication typically uses distributed sensors that send DNS replies to a global database, stores DNS records of all types, and provides interfaces for querying such records. RUS-CERT at the University of Stuttgart maintains such a global database and offers a restricted query front end [17].

IV. Design and implementation

This section describes DNS Flagger.

A. Architecture

DNS Flagger's goal is to identify bot-infected hosts in a *monitored network*. The monitored network may, e.g., comprise an ISP's subscribers in a metropolitan region, as illustrated in Fig. 1. DNS Flagger is installed in a *service network* that also contains the ISP's DNS resolvers and DHCP servers. We assume that (1) all DNS traffic between the monitored network and the Internet passes through this service network, (2) all dynamic IP address allocation by the ISP is performed by the service network's DHCP servers, and (3) botmasters can corrupt any node or link in the Internet or in the monitored network, but not in the service network.

The service network should be configured such that a copy of each DNS query and reply is sent to DNS Flagger. This copying and forwarding may be performed by a switch, router, or middlebox between the monitored and service networks.

DNS Flagger verifies whether any monitored host's DNS queries match a pattern indicative of bot infection. However, a DNS packet will contain only the monitored host's IP address. If monitored hosts use dynamically allocated IP addresses, host identification by IP address could be equivocal. To avoid confusing hosts that use the same IP address at different times, DNS Flagger should also receive a copy of the packets used for IP address allocation by the monitoring organization (e.g.,

DHCP acknowledgements from DHCP servers and declines and releases from monitored hosts). For example, a switch, router, or middlebox between the monitored and service networks may be configured to forward copies of these DHCP packets to DNS Flagger.

If the monitored network is not very large, copies of all packets that transit between the monitored and service networks may be sent to DNS Flagger. This can be accomplished, in increasing order of capacity, by a hub, a switch configured for port mirroring, or a network tap. On the other hand, if the monitored network is very large, a router or middlebox with the ability to filter and forward to DNS Flagger only copies of DNS and DHCP packets should be used, such that DNS Flagger receives and processes only relevant packets.

A botmaster could attempt to disrupt botnet detection by sending from compromised nodes spoofed DNS or DHCP packets. To block spoofed packets, the service network should perform ingress filtering [8]. Packets with source address in the monitored or service network but arriving instead from the Internet should be dropped (e.g., by a firewall between the service network and the Internet). Likewise, packets arriving from a monitored host with source address not assigned to that host should be dropped (e.g., by an access router, firewall, or switch between the monitored and service networks). Techniques such as DHCP relay agent information option [14] and DHCP snooping [5] may be used for such filtering. Additionally, the monitoring organization's DNS resolvers should be configured to avoid DNS cache poisoning [19]. Given that DNS Flagger is a passive monitoring device, it detects and filters out DNS cache poisoning only internally.

B. Configuration, updates, and usage

DNS Flagger is configured with a *blacklist* containing the domain names and/or IP addresses of known botnet C&C servers. A host that issues a DNS address query containing a domain name in the blacklist, or receives a DNS address reply containing an IP address in the blacklist is presumably infected by a bot. DNS Flagger ordinarily generates an alarm using syslog in such cases. However, a *whitelist* may contain the fixed IP address, MAC address, or client/remote identifier [14] of each monitored host for which alarms should not be generated. There is no need to include in a whitelist hosts in the service network, because only monitored hosts can cause alarms. After a host's query triggers an alarm, DNS Flagger suppresses further alarms for the same host.

DNS Flagger's blacklist needs to be updated regularly, similarly to virus definitions in AVs. Therefore, DNS Flagger should be configured with a URL and initial time and period for picking up blacklist updates. The URL may point, for example, to the site of a vendor or alliance that makes such updates available, as explained in Section II.

To obtain other names of C&C servers in its blacklist, DNS Flagger uses local DNS signature harvesting. DNS Flagger automatically updates its blacklist when it observes a reply to a DNS address query associating a new domain name to an IP address already in the blacklist. DNS Flagger proactively attempts to trigger such replies by performing a reverse DNS lookup for each new IP address in its blacklist update, and then issuing a corresponding DNS address query. If the reply to the latter is consistent (i.e., gives the blacklisted address), the corresponding name is entered in the blacklist.

Additionally, DNS Flagger may be configured with a URL, initial time, period, and scope for consulting or updating a global passive DNS replication database. The scope defines what C&C servers are included in DNS Flagger's queries. The scope may be restricted to C&C servers picked up in the most recent update from the vendor or alliance, or, on the contrary, include all C&C servers in the blacklist. If DNS Flagger is configured for updating, it sends to the global database any new information harvested from local DNS traffic.

DNS Flagger does not by itself affect the operation and performance of hosts and networks. In particular, DNS Flagger does not blacklist any hosts. Administrators need to follow up on DNS Flagger's alarms and have the affected hosts possibly quarantined and disinfected.

C. Implementation

We implemented a DNS Flagger prototype on a Dell Dimension 9200 computer with Intel Core 2 Duo CPU at 2.4 GHz, 2 GB RAM, and Gigabit Ethernet interface. We wrote the DNS Flagger program in C using the pcap library on Linux 2.6. The program performs stateful deep-packet inspection to avoid poisoning [19] and false positives [15], and uses hash tables extensively to reduce search costs.

V. Measuring bot detection rate

This section describes our methodology for measuring bot detection rates.

Our setup uses a sandbox based on Truman [20]. We modified Truman's gateway in four ways. First, we added firewall rules such that sandbox hosts can communicate with the Internet (but not other campus hosts), with limited bandwidth (56 Kbps) and limited number of concurrent sessions. These rules allow sandbox hosts to receive commands from C&C servers on the Internet, but restrict their involvement in DDoS attacks. (We kept Truman's SMTP sink so that spam does not emanate from the sandbox.) Second, we added to the gateway a proxy DNS resolver for sandbox hosts. Third, we modified the gateway such that it forwards a copy of the relevant network traffic to DNS Flagger. Fourth, we modified the gateway such that it can run unattended. The gateway continuously performs the following actions:

1) Every full hour, download all malware specimens freshly collected by MWCollect honeypots in the previous hour.
2) Forty five minutes after the hour, submit each specimen collected above to VirusTotal.
3) If any of VirusTotal's 32 different AVs classify a specimen as a bot, record the AVs' classifications, load the specimen on a sandbox host, run it for 10 minutes, and record DNS Flagger's alarms (if any).

The information recorded by the gateway then allows us to compute the AVs' and DNS Flagger's bot detection rates.

Note that, for realism, the specimens used for testing are *not* historical samples, but rather specimens that are actively

TABLE I
BOT DETECTION RATES

Defense	Bot detection rate (%)			
	by itself	w/ IP signatures	w/ local DNS signatures	w/ global DNS signatures
DNS Flagger		**53.9**	**69.2**	**73.1**
Symantec	65.4	84.6	88.5	92.3
McAfee	61.5	80.8	84.6	86.5
Sophos	44.2	71.2	82.7	86.5
Panda	65.4	78.9	92.3	94.2
F-Secure	67.3	80.8	92.3	96.2
Norman	59.6	75.0	88.5	92.3
AhnLab-V3	55.8	73.1	86.5	90.4
Kaspersky	63.5	80.8	92.3	96.2
Microsoft	61.5	73.1	86.5	88.5
ClamAV	46.2	73.1	82.7	86.5
AV average	**59.0**	**77.1**	**87.7**	**91.0**
AV std. dev.	8.0	4.6	3.8	3.9

propagating and were caught in MWCollect honeypots up to an hour before. Moreover, the sandbox runs and includes in experimental results *all* specimens that are classified as a bot by *any* of VirusTotal's AVs. Experimenters do not know beforehand what bots will be tested and cannot bias sample selection. Note also that MWCollect timestamps and publicly archives all specimens considered, facilitating reproduction of experimental results.

VI. EXPERIMENTAL RESULTS

This section reports DNS Flagger's measured detection and false alarm rates and throughput.

A. Detection rate

We used the method described in Section V on three randomly selected consecutive days. During those days, MWCollect honeypots collected a total of 439 malware specimens. All 439 specimens were found to be malware and 52 of them received a bot label from at least one of VirusTotal's AVs. Kaspersky AV classified the bots into the Rbot (17), Sdbot (15), Bobic (5), Virut (5), Sality (3), Sramler (2), Agent, Allaple, Delf, EggDrop, and IRCBot families. Other AVs assigned these bots a variety of other labels, including Agobot, Bobax, Linkbot, MyBot, Mytob, PoeBot, Robobot, Rxbot, Spybot, Vanbot, Virtob, and Woobot. Three of the 52 bots were duplicates (i.e., our sample contained 49 different executables, achieving sample size, randomness, and freshness exceeding those of many previous works). DNS Flagger was configured to pick up C&C IP signatures from Shadowserver daily at 1:35 a.m. and to consult RUS-CERT's passive DNS database immediately thereafter, querying only about new addresses.

We analysed what bots DNS Flagger was able to detect (1) using only Shadowserver IP signatures, (2) using only locally harvested DNS signatures but not IP signatures, or (3) using only DNS signatures from the global passive DNS database but not locally harvested DNS signatures or IP signatures. We considered that DNS Flagger detected a bot if DNS Flagger generated an alarm when the bot ran in the sandbox for 10 minutes. The results are shown in Table I. Using only IP signatures, DNS Flagger detected only 53.9% of the bots. Locally harvested DNS signatures significantly improved bot detection rate, to 69.2%. Queries to the global passive DNS database gave a smaller additional improvement to 73.1%.

We also measured which bots each AV detected. We considered that an AV detected a bot if the AV found the specimen to be any form of malware (not necessarily a bot). Table I shows the performance of major AVs [12]. (Trend Micro and Computer Associates are also major AVs; they are missing in the table because VirusTotal does not provide results for them. On the other hand, the table also includes results for Microsoft's AV and ClamAV, a popular open-source AV.) Major AVs' average bot detection rate was only 59%. Trade magazines often quote much higher detection rates for these AVs. That is because they test AVs with large repertoires comprising mostly historical malware, for which signatures are likely to exist. On the contrary, our tests are with malware actively propagating and caught in the wild in the previous two hours.

We then calculated the combined bot detection rate using each major AV and DNS Flagger. The results are also shown in Table I. Combining a major AV with detection based on IP signatures on average improved bot detection rate from 59% to 77.1%. Combining the latter with detection based on locally harvested DNS signatures significantly improved bot detection rate, to 87.7%. Finally, combining the latter with DNS signatures from the global passive DNS database gave a smaller further improvement, to 91%.

We also analyzed why DNS Flagger failed to detect certain bots during the experiment. Bots queried DNS for the address of a C&C server that was not in the blacklist in 9.6% of cases. None of the missing servers were included in Shadowserver's blacklist until more than two weeks after the experiment. Thus, simply increasing update frequency from daily to hourly would not have improved the detection rate. About 5.8% of the bots

used a fixed C&C server address and thus cannot be detected based only on DNS traffic. Another 5.8% of the bots did not contact a C&C server and simply scanned other hosts or did not communicate at all. It is possible that some of these might wait longer than ten minutes to start communicating with a C&C server, and could have been detected if allowed to run longer in the sandbox. About 5.8% of the bots were not detected because of temporary DNS server failure. This was an experimental artifact that could have been avoided by configuring alternate DNS resolvers in the sandbox's gateway. When we ran these bots again with the same blacklist, DNS Flagger detected them. Without this glitch, the average bot detection rate of a major AV plus DNS Flagger with all signatures would have been 94.2%.

B. False alarm rate

To evaluate whether DNS Flagger would issue false alarms in production, we connected it to our department's network during three randomly selected contiguous work days. The network comprises slightly more than 400 hosts. To ensure that there would be at least some alarms, we also ran four bots (Bobax and Sdbot variants) in our sandbox and sent their DNS traffic to DNS Flagger.

DNS Flagger generated alarms for all four bots in the sandbox and for two additional hosts from the department. After investigation, we found that one of the latter was a retired but still connected server running an unpatched old version of Linux (Red Hat 8). Network logs clearly indicated that it was being used as a bot. The second host from the department turned out to be a student's notebook computer running Windows XP. After updating its AV, the AV identified a piece of malware running in the computer. There were no false alarms.

C. Throughput

To measure DNS Flagger's throughput, we connected our prototype to four other computers that replayed DNS traces at a controllable rate. We measured how many DNS packets one of DNS Flagger's CPU cores was able to process as the packet arrival rate varied. There were 1336 domain names in DNS Flagger's blacklist. DNS Flagger's throughput on a single core peaked above 250 Kpackets per second.

To estimate the size of the population that could be monitored by DNS Flagger, we analyzed a trace containing all DNS traffic in our department for nine randomly selected contiguous days. A total of 409 distinct hosts belonging to the department can be identified in the trace. In every second throughout the trace's nine days, the number of DNS packets was less than the total number of hosts in the department. These results suggests that, if it receives only DNS and DHCP packets, DNS Flagger could handle more than 100,000 subscribers on each of its CPU cores.

VII. Related Work

Our results suggest that bot detection based on IP and DNS signatures can provide trade-offs between detection rate, false alarm rate, and throughput that would be of value to many subscribers and ISPs. Nonetheless, there don't seem to be previous reports evaluating such an approach or comparing it to major AVs.

Most research on bot detection has targeted campus or enterprise networks, which typically are much smaller than are ISP networks. Whyte et al. proposed a scheme based on DNS anomalies for detecting malware propagation [24]. Their scheme attempts to detect communication whose initiator did not previously query DNS for the responder's address. Such a communication pattern is typical of infected hosts that choose propagation targets randomly, using numerical address calculations. Unlike DNS Flagger, Whyte et al.'s scheme needs to process all traffic and depends on a well-maintained whitelist to reduce false alarms, because there are many exceptions (e.g., legitimate communication with network infrastructure nodes often uses hardcoded addresses).

A more recent DNS-based scheme was proposed by Choi et al. [4]. Their scheme seeks to detect groups of DNS queriers with characteristics suggestive of bot infection: fixed group membership, synchronized queries, and use of dynamic DNS. However, the complexity of algorithms for group detection limits their scalability. BotSniffer is another scheme based on detection of groups with synchronized network traffic [10]. BotMiner extends this approach to detect P2P botnets [11]. Group-based approaches need the monitored network to be large, such that it is likely to contain more than one bot from any given botnet, but not so large that algorithm runtime becomes excessive. For example, in the experiment described in Section VI-B, DNS Flagger unexpectedly detected two singleton bots in our department's network. Because they belonged to different botnets, these bots could not have been detected with a group-based approach.

Binkley and Singh proposed an algorithm for detecting bots that use the IRC protocol for C&C [3]. They define a host's work weight as the ratio between the number of TCP SYNs and FINs sent and RSTs received by the host to the total number of TCP packets sent or received by the host. Hosts with high work weight are suspected of harboring malware that is scanning for propagation. IRC channels with many such hosts are classified as evil. The algorithm detects bots only after attacks are well under way, and may generate false alarms. Binkley and Singh recommend that their algorithm be used together with a signature-based scheme that enables earlier detection of bots with known signatures. DNS Flagger would fit this role well.

Like Binkley and Singh's algorithm, Rishi monitors IRC traffic to detect bots [9]. Rishi heuristically measures the similarity between payloads of each TCP connection and messages of known IRC botnets. In particular, Rishi detects known IRC commands and evaluates the similarity of IRC nicknames. Based on the similarity measures, Rishi calculates how likely each host is to be infected. Rishi can detect unknown bots earlier than Binkley and Singh's algorithm, but also may generate false alarms and thus requires manual filtering by administrators. Unlike Rishi, DNS Flagger can detect bots that use encrypted messages or C&C protocols other than IRC. Because DNS Flagger needs to process only

DNS and DHCP packets, it also scales more easily.

Ramachandran et al. have proposed a bot detection method based on DNS blackhole list (DNSBL) counter-intellingence [16]. ISPs and enterprises often reject email from senders in DNSBLs, which contain known spammers. Therefore, attackers prefer to send spam from unlisted bots. Ramachandran et al. found evidence that botmasters use listed bots to check in DNSBLs whether other bots are still unlisted. Thus, if a host is the target of a DNSBL query by a listed bot, there is a good chance that the target is also infected. However, their scheme may generate false alarms and would be best deployed by DNSBL providers. On the contrary, DNS Flagger gave no false positives in our tests and can readily be installed in ISPs.

VIII. Conclusions

Many home subscribers currently lack protection against bot infection. Infection hurts host performance, jeopardizes privacy, and enables identity theft and other attacks. It also causes collateral damage to ISPs and other subscribers in the form of clogged network links and blacklisted or greylisted email servers. In order to provide value to subscribers, an ISP bot detection service should offer detection rates at least as good as those of major AVs, and substantially better combined detection rates for hosts that already have a major AV. At the same time, an ISP bot detection service must readily scale to hundreds of thousands of subscribers. We described DNS Flagger, a novel device that satisfies these requirements. Because its algorithms are simple, a single DNS Flagger can handle an ISP's entire metropolitan area. Yet, DNS Flagger bot detection rates significantly improve on those of major AVs. And, like the latter, DNS Flagger exhibited no false alarms in our experiments. DNS Flagger is easy to configure on the ISP side and does not require any subscriber-side installation or reconfiguration. DNS Flagger can use readily available IP signatures, from which it derives DNS signatures that greatly improve performance. Many previous proposals target campus or enterprise networks, which are much smaller. Nonetheless, their bot detection rates cannot be much greater than that of DNS Flagger combined with a major AV (which is greater than 90%). Considering not only detection and false alarm rates but also simplicity and low cost, many subscribers and ISPs may find DNS Flagger a compelling network service. A bot detection system of similar approach or scalability does not seem to have been evaluated in previous literature.

Acknowledgements

This project was funded in part by The Technology Collaborative through a grant from the Commonwealth of Pennsylvania, Department of Community and Economic Development.

References

[1] P. Bächer, T. Holz, M. Kötter and G. Wicherski. "Know Your Enemy: Using Honeynets to Learn More About Bots." [Online] http://www.honeynet.org/papers/bots/

[2] P. Bächer, M. Kötter, T. Holz, M. Dornseif and F. Freiling. "The Nepenthes Platform: An Efficient Approach to Collect Malware," in *Proc. 9th Intl. Symp. on Recent Advances in Intrusion Detection (RAID)*, LNCS 4219, Springer-Verlag, Sept. 2006, pp. 165-184.

[3] J. Binkley and S. Singh. "An Algorithm for Anomaly-Based Botnet Detection," in *Proc. 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, USENIX, July 2006, pp. 43-48.

[4] H. Choi, H. Lee, H. Lee and H. Kim. "Botnet Detection by Monitoring Group Activities in DNS Traffic," in *Proc. 7th Intl. Conf. Computer and Information Technology*, IEEE, 2007, pp. 715-720.

[5] Cisco. "Understanding and Configuring DHCP Snooping." [Online] http://www.cisco.com/en/US/docs/switches/lan/catalyst4500/12.1/13ew/configuration/guide/dhcp.html

[6] CNN. "Expert: Botnets are Number One Internet Threat." CNN.com, Jan. 31, 2007. [Online] http://www.cnn.com/2006/TECH/internet/01/31/furst/

[7] E. Cooke, F. Jahanian and D. McPherson. "The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets," in *Proc. Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, USENIX, July 2005, pp. 39-44.

[8] P. Ferguson and D. Senie. "Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing," IETF, RFC 2827, May 2000.

[9] J. Goebel and T. Holz. "Rishi: Identify Bot Contaminated Hosts by IRC Nickname Evaluation," in *Proc. 1st Workshop on Hot Topics in Understanding Botnets*, USENIX, April 2007.

[10] G. Gu, J. Zhang and W. Lee. "BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic," in *Proc. 15th Annual Network and Distributed System Security Symp.*, ISOC, Feb. 2008.

[11] G. Gu, R. Perdisci, J. Zhang and W. Lee. "BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection," in *Proc. 17th USENIX Security Symp.*, July 2008.

[12] A. Hallawell and P. Firstbrook. "Magic Quadrant for Enterprise Antivirus, 2006." Gartner RAS Core Research Note G00141873, Aug. 2006. [Online] http://www.mcafee.com/us/local_content/misc/2006_av_mq.pdf

[13] MWCollect. "Malware Dedicated Whitehats." [Online] http://www.mwcollect.org/

[14] M. Patrick. "DHCP Relay Agent Information Option," IETF, RFC 3046, Jan. 2001.

[15] S. Patton, W. Yurcik and D. Doss. "An Achilles' Heel in Signature-Based IDS: Squealing False Positives in SNORT," in *Proc. 4th Intl. Symp. on Recent Advances in Intrusion Detection (RAID)*, Oct. 2001.

[16] A. Ramachandran, N. Feamster and D. Dagon. "Revealing Botnet Membership Using DNSBL Counter-Intelligence," in *Proc. 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, USENIX, July 2006, pp. 49-54.

[17] RUS-CERT. "Passive DNS Replication." [Online] http://cert.uni-stuttgart.de/stats/dns-replication.php

[18] Shadowserver Foundation. [Online] http://www.shadowserver.org/wiki/pmwiki.php?n=Shadowserver.Shadowserver

[19] U. Steinhoff, A. Wiesmaier and R. Araújo. "The State of the Art in DNS Spoofing," in *Proc. 4th Intl. Conf. Applied Cryptography and Network Security (ACNS),* Industrial Track, June 2006.

[20] J. Stewart. "Truman - The Reusable Unknown Malware Analysis Net." [Online] http://www.secureworks.com/research/tools/truman.html

[21] VirusTotal. "Free Online Virus and Malware Scan." [Online] http://www.virustotal.com/

[22] Y. Wang, D. Beck, X. Jiang, R. Roussev, C. Verbowski, S. Chen and S. King. "Automated Web Patrol with Strider HoneyMonkeys: Finding Web Sites that Exploit Browser Vulnerabilities," in *Proc. Network and Distributed Systems Security Symp. (NDSS)*, ISOC, Feb. 2006.

[23] F. Weimer. "Passive DNS Replication," in *Proc. 17th Annual FIRST Conf.*, July 2005. [Online] http://www.first.org/conference/2005/papers/florian-weimer-paper-1.pdf

[24] D. Whyte, E. Kanakis and P. van Oorschot. "DNS-based Detection of Scanning Worms in an Enterprise Network," in *Proc. 12th Annual Network and Distributed System Security Symp.*, ISOC, Feb. 2005.