

The Days After a “\0” Scan from the Sality Botnet

Elias Raftopoulos
ETH Zurich
Zurich, Switzerland
riliias@tik.ee.ethz.ch

Eduard Glatz
ETH Zurich
Zurich, Switzerland
eglatz@tik.ee.ethz.ch

Xenofontas Dimitropoulos
FORTH-ICS
Crete, Greece
fontas@ics.forth.gr

Alberto Dainotti
CAIDA-University of California
San Diego, US
alberto@caida.org

Abstract—Although Internet scanning is one of the most popular malware propagation methods, sound measurements about its success rate are not generally available. In this work, we assess the success rate of an Internet-wide scanning event that was orchestrated by the Sality botnet during February 2011 using data from a university network. We first use unsampled NetFlow records from the border router of the network to find how many targeted hosts replied to the scanners. Second, we correlate the replies with IDS alerts triggered in the same network and uncover significant exploitation activity that followed toward the scan repliers. In our data, 2% of the scanned hosts replied and at least 8% of the repliers we believe were eventually compromised. Furthermore, we characterize the exploitation activity and find surprisingly that scanners and exploiters came from different geographical locations. Our analysis provides a novel look into the success rate of Internet scanning in the wild based on two unique data-sets.

Keywords—Botnet Characterization, Network Forensics, Network Scanning, IDS, Netflow

I. INTRODUCTION

Botnets of up to millions of compromised computers are presently the most widely-used cyberweapon for executing criminal activities, such as fraud, sensitive data leakage, distributed denial-of-service attacks, and spam. Botnets engage into large-scale scanning to enumerate vulnerable hosts for targeted criminal activities or simply propagation [1], [2]. A recent study showed that scanning accounts for 34-67% of all connections (successful and unsuccessful) in an ISP [3]. Besides, recent advances in scanning software make possible to scan the entire IPv4 address space in less than 45 minutes [4], simplifying further the execution of aggressive scanning attacks. In spite of the prevalence of scanning, measurements about its success rate in the Internet are not generally available. Network administrators often ignore scanning as a baseline noise that does not pose a significant threat.

In this work, we use unsampled Netflow records and IDS alerts collected from a university network to assess the success rate of an Internet-wide scanning event. In particular, we focus on the “sipscan”, an Internet-wide scanning event orchestrated from the Sality botnet over 12 days in February 2011 that was uncovered by Dainotti *et al.* [5]. This event had several unique characteristics over previously known scanning attacks: 1) it used a well-orchestrated stealth scanning strategy; 2) it originated from 3 million IP addresses; 3) it is believed that it scanned the entire Internet address space; and 4) it was targeted towards Session Initiation Protocol (SIP) [6] servers.

We discovered that the scanning event escalated into persistent exploitation attempts towards the hosts that replied

to the sipscan. We use our data to assess the effectiveness of scanning in terms of scan repliers and hosts that were eventually compromised. We find that 2% of the scanned IP addresses replied and at least 8% of the repliers were eventually compromised. Besides, our analysis shows that scanners originated primarily from Eastern countries, while the subsequent exploitation attempts originated from Western countries. This suggests that information about scan repliers was communicated to the subsequent attackers likely through underground channels. Moreover, we observe 352,350 new scanner IP addresses and show that the sipscan was largely undetected by the IDS, which only raised alerts for 4% of the scan probes.

In summary, our work makes the following **contributions**:

- We assess the effectiveness of an Internet-wide scanning event. To the best of our knowledge, this is a first measurement study that focuses on Internet scan repliers.
- We uncover and characterize how an interesting /0 scan escalated to exploitation activity. Among other observations, our analysis shows that the subsequent exploitation attempts came from different geographical locations and that the sipscan originated from 352,350 more IP addresses than previously seen.

The rest of the paper is structured as follows. We first discuss related research in Section II. In Section III we describe the used data-sets. Then, Section IV presents how unsampled NetFlow records were used to detect the sipscan and measure scan repliers. Then, in Section V we characterize the exploitation activity that followed based on our IDS data. Finally, Section VI discusses the impact of false-positive IDS alerts on our analysis and Section VII concludes our paper.

II. RELATED WORK

A long line of measurement studies has analyzed botnets over the last years, following their evolution from centralized IRC-based [7], [8] to fully decentralized C&C architectures [9], [10]. The goal of these efforts has been to characterize botnet activities [11], analyze C&C communication methods [7], and estimate the respective botnet size and geographical properties [12]. Their observations have been used to fine tune network defences [13] and tailor novel detection mechanisms [14], [15], [16].

One of the most integral aspects of botnet activity is scanning. Since scanning is widespread [3] and regularly captured by monitoring infrastructures [17], [8], it is imperative

for security analysts to have a measure regarding its severity and impact on the victim population. However, few studies have focused on the probing characteristics of botnets. In [18] Paxson *et al.* analyzed traffic captured at honeynets in order to study the statistical properties of 22 large-scale scanning events. In a followup study, Li *et al.* [19], [20] extracted botnet scan traffic from honeynet data and used it to infer general properties of botnets, such as population characteristics, black-listing effectiveness, dynamics of new bot arrivals and scanning strategies. Finally, Yegneswaran *et al.* [8] analyzed the source code of a widely-used botnet malware, revealing the scanning capabilities of basic IRC bots.

Most related to our work, Dainotti *et al* [5] discovered an interesting stealthy scan of the entire IPv4 address space that was carried out by the Sality botnet and analyzed the different phases of the event. However, this study was based solely on packet traces collected at the UCSD network telescope and does not provide insights regarding the effectiveness of scanning and its followup activity. In our work, we detect the sipscan in a large ISP with live hosts, identify the set of hosts that replied to scanners, and analyze the targeted exploitation activity that followed. This way provide new insights about the escalation of this event and the effectiveness of scanning.

III. MONITORING INFRASTRUCTURE AND DATA COLLECTION

In this section, we describe the monitored network and the data we use in this study. We collected our measurements from the network of the main campus of the Swiss Federal Institute of Technology at Zurich (ETH Zurich). The ETH Zurich network is large and diverse. During our data collection period, which spanned 5 months (between the 1st January and the 31th of May 2011), we observed in total 79,821 internal hosts. On these hosts, the IT policy grants full freedom to users regarding the software and services they can use.

We select two data sources that provide complementary views into the studied event. Firstly, we collect unsampled NetFlow data from the only upstream provider of ETH Zurich. Netflow produces summary records for *all* flows crossing the monitoring point. However, Netflow lacks context, since it does not provide information regarding the type of activity that triggered a flow. To fill this gap, we use IDS data collected from a Snort sensor, which captures and analyzes all traffic crossing our infrastructure's border router. Snort uses signature-based payload matching to perform protocol analysis, revealing more information about the type of activity that triggered an observed packet sequence. The two passive monitoring tools complement each other, since they capture flow summaries for all traffic and finer (payload/header) details for packets that trigger IDS signatures.

A. Netflow Data

We use unsampled NetFlow records collected at SWITCH, a regional academic backbone network that serves 46 single-homed universities and research institutes in Switzerland [21]. The hardware-based NetFlow meters capture all the traffic that crosses the border destined to or coming from the Internet. In a single peering link, we observe in 2011 on average 108.1 million flows per hour, which correspond to 3,064 million

packets. From each flow record we extract the following fields: IP addresses, port numbers, protocol number, byte/packet counts, and timestamp. We do not use TCP flags because they are not supported by the fast hardware-based NetFlow. We dissect NetFlow records into one- and two-way flows using a number of preprocessing steps, which are described in detail in [3]. We first assign each flow to a time interval by its start time, then defragment flows which have been fragmented into multiple NetFlow records, and finally pair two-way flows. For TCP and UDP, a two-way flow is the aggregate of two defragmented flows that have the same 5-tuple with reverse values in the source and destination IP address and port fields. Accordingly, a one-way flow is a flow that does not have a matching reverse flow. We search for matching one-way flows in the same and adjacent time intervals. In [3] we also describe how we eliminate double-counting of flows and handle the problem of asymmetric routing. For this study, we focus on those flows that involve the IP address range allocated to ETH Zurich. We analyze flow data for the first 400 hours of February 2011 which is equivalent to 16.7 days.

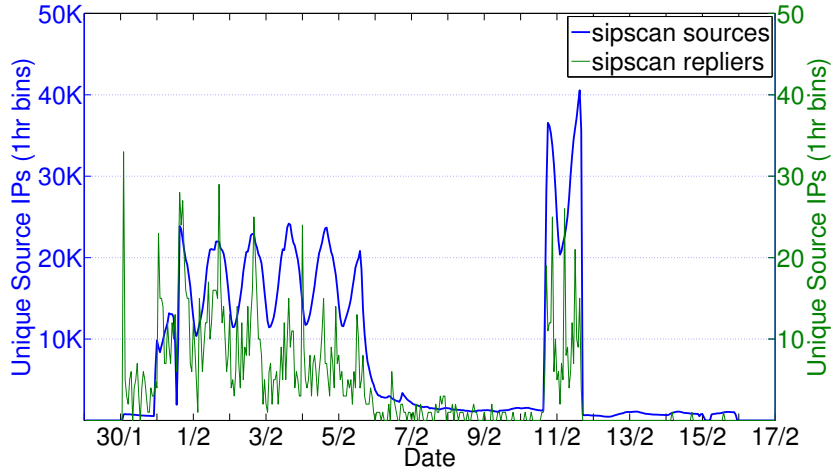
B. IDS Alerts

We collect alerts generated by the popular Snort IDS in the border router of the monitored network. The IDS monitors both downstream and upstream traffic and triggers an alert when a packet matches a signature. Snort is configured with the two most widely-used signature rulesets, namely the Vulnerability Research Team (VRT) and the Emerging Threats (ET) rulesets. The former is the default Snort ruleset, which accounts for 5,559 rules, developed and maintained by Sourcefire. The latter is an open-source ruleset, which is maintained by contributors and accounts in total for 11,344 rules. By using both rulesets, our Snort sensor is sensitive to a wide range of possibly malicious events, including reconnaissance scanning, active attacks, malware spreading and communication, data leakage and host compromise.

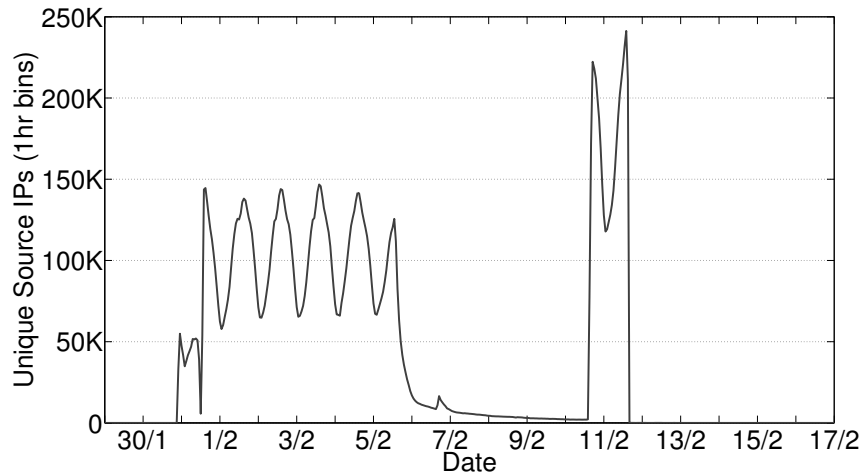
The collected alerts have the standard full Snort format. For example, the following is a medium priority scan alert, that was triggered by SIPVicious inbound traffic. The SIPVicious suite is a set of tools that can be used to enumerate and audit SIP-based VoIP systems. The IP addresses in this example have been anonymized:

```
[**] [1:2011766:4] ET SCAN Sipvicious User-Agent Detected [**]  
[Classification: Attempted Information Leak] [Priority: 2]  
01/01-01:57:00.272686 a.b.c.d:12312 -> m.n.p.r:5060  
UDP TTL:48 TOS:0x0 ID:0 IpLen:20 DgmLen:439 DF Len: 411
```

We have tailored a Perl module that parses raw IDS alerts and extracts a set of features relevant to our analysis. Specifically, the fields that we use are the rule identification number (2011766), the rule description (ET SCAN Sipvicious User-Agent Detected) and classification (Attempted Information Leak) that provides the context of the security event, the rule priority which is a severity measure, the event timestamp (01/01-01:57:00.272686), and the involved IP addresses and port numbers (a.b.c.d, 12312, m.n.p.r, 5060). In parenthesis we illustrate the respective extracted values for the Sipvicious scan example shown above.



(a) ETH Zurich (NetFlow trace)



(b) UCSD network telescope (full packets trace)

Fig. 1: Number of IP addresses per hour sourcing or replying to scan flows in ETH Zurich and in the UCSD network telescope.

IV. SIPSCAN DETECTION

To extract sipscan traffic from NetFlow data, we rely on heuristics introduced by Dainotti *et al.* [5], which are based on the analysis of the payload of sipscan packets. However, because flow data do not include packet payload contents, we adapted the extraction rules. We focus on the UDP part of sipscan traffic, which is sufficient to detect sipscan activity and identify sipscan sources. Specifically, we identify a sipscan flow as a single-packet one-way flow towards port 5060/udp having a size in the range of 382 to 451 bytes.

In Figure 1a, we highlight how the host population sourcing attacks towards the SIP service port evolved over 16.7 days (from 31/01/2011 to 16/02/2011). In Figure 1b, we illustrate how the same event was captured by the UCSD network telescope. Note that Dainotti *et al.* [5] used full packet traces collected at the network telescope in order to estimate the scanning population. The similarity in these two patterns, indicates that our heuristics adapted to Netflow records are able to capture the same phenomenon as seen on our network. We observe two major sipscan outbreaks in terms of participating

attackers along with a minor fraction of hosts engaged continuously in SIP scanning. The first outbreak starts at 2011.01.31 21:30 UTC and lasts until approximately 2011.02.06 22:40, while the second outbreak starts at 2011.02.11 14:10 and lasts until 2011.02.12 15:00 UTC. In total, 952,652 scanners participated in the scan. A significant number (352,350) of hosts targeting our infrastructure were not observed in the population of Sality scanners detected by the UCSD network telescope, which were 2,954,108 [5]. This finding indicates that the size (expressed in terms of source IP addresses) of the botnet was at least 11.9% larger than the lower bound estimated in the previous work.

At the victim side, 77,158 hosts within ETHZ were scanned at least once during the 16.7 days period, meaning that the coverage of the scan in our infrastructure was 96.6%. The scan was largely stealthy, in terms of generated alerts from the IDS, since only 4% of the respective probing flows triggered a scan-related IDS signature.

In contrast to [5], our data set allows us to identify those target hosts that reply to the sender of a sipscan flow. For this

purpose, we search for two-way flows matching a relaxed filter (i.e., requiring port 5060/UDP only). Additionally, we look at the number of attacker-victim host pairs where a sipscan flow is answered with an ICMP flow. For this answer type, we see a weak correlation of ICMP flow counts with the two sipscan outbreaks. On the other hand, when looking at host pairs where we have biflows, we observe a strong correlation of biflow counts with the sipscan outbreaks indicating that sipscan attacks significantly result in bidirectional communication between attacker and victim. In Figure 1a we present the number of unique internal IP source addresses responding to the sipscan. In total, we identify 1,748 sipscan repliers, whereas during the scan we find 3.8 new unique internal IPs responding to the scan every hour. For 80.2% of the repliers we detected a TCP reply originating from the respective host, whereas for 8.3% of the repliers, the sipscan was answered with an ICMP flow. 0.2% of the replies involved both a TCP and an ICMP flow, while the remaining 11.5% used neither TCP or ICMP.

V. AFTERMATH OF THE SIPSCAN

A. Inbound exploitation attempts

In this section, we study the impact of the sipscan on the target host population within ETH Zurich. We first investigate if scanning was a precursor of subsequent exploitation attempts targeting hosts that replied to the scanners. Recall that our IDS data cover 5 months, including one month before the beginning of the sipscan (31/01/2011) and approximately 3.5 months after its end (16/02/2011).

In Figure 2, we show how the daily number of exploitation alerts per target host triggered by inbound traffic changed after the sipscan. We consider alerts of the VRT rulesets *exploit.rules*, *exploit-kit.rules*, and *indicator-shellcode.rules* and of the ET ruleset *emerging-exploit.rules*. These rulesets are tailored to detect exploit activity, including buffer overflow attacks, remote command execution, brute force authorization and privilege escalation attempts. In Figure 2, we also show the daily number of exploitation alerts per target host for the *baseline*, i.e., the ETH Zurich hosts that did not reply to the scanners according to our data. The *baseline* accounts for 78,073 hosts, whereas the number of sipscan repliers is 1,748. In the pre-sipscan period sipscan repliers were involved on average in 122 exploitation alerts per day. During the sipscan period we see that this number increases to 842 alerts per day, whereas after the sipscan it remains high at 931 alerts per day. In sharp contrast, the inbound exploitation activity associated with the *baseline* remains low after the sipscan. On average, each host is a target in 1.2 alerts per day, which is a baseline noise caused by automated threats attempting to propagate and false alerts. The respective noise level for the sipscan repliers in the pre-sipscan period is 0.4 alerts per day. After the sipscan, this number increases to 3.7 alerts per day. The high number of exploitation alerts towards sipscan repliers persists even 4 months after the end of the sipscan, although it is more intense during the first two months (from 31/1 to 28/2), when 68% of the total exploitation alerts are triggered. Out of the 1,748 sipscan repliers, we observe that 852 were involved in inbound exploitation alerts.

In addition, we explore how persistent the attacking hosts are in terms of generated exploitation alerts, and examine

whether the attackers targeting the sipscan repliers are more persistent compared to the ones targeting the baseline. In Figure 3, we compare the average number of exploitation alerts per target for sipscan repliers and baseline attackers, respectively. We see that the former group tends to be more persistent triggering in the median case 4 exploitation alerts per target, whereas the same number for the latter group is 2 alerts. The increased persistence towards sipscan repliers is more prominent in the tails of the distributions. We see that the top 10% most active attackers towards sipscan repliers launch up to 73 alerts on average per target, whereas the respective number for the baseline is only 21 alerts.

Next, we study whether the observed increase in exploitation activity comes from new offenders. Figure 4 illustrates the daily number of new offending IP addresses per target host for sipscan repliers and for the baseline. We report IP addresses that appear in exploitation alerts, however we consider an address new only when it has not previously appeared in the entire alert trace. A baseline host records a new external attacker approximately every four days consistently throughout the 5-month period. However, this number increases sharply for sipscan repliers during the sipscan, when each victim is attacked on average by 1.4 new IP addresses per day. Moreover, we investigate whether these IP addresses are known blacklisted hosts using four public blacklists [22], [23], [24], [25]. Figure 4 shows that only 7% of the new offenders were already blacklisted, while this number drops to almost 0 before and after the sipscan period.

We also investigate the similarity between the IP addresses of scanners (extracted from NetFlow) and of exploiters (extracted from Snort alerts towards sipscan repliers). Surprisingly, we observe that out of 6,676 exploiter and 1.3 million scanner IP addresses, only 17 are in common. This suggests that there is a clear separation between scanners and bots wielded to exploit target hosts. In Table I, we compare the geographical distribution of the scanners detected in our infrastructure and in the UCSD network telescope [5] with the exploiters targeting the ETH Zurich sipscan repliers. The geographical distribution of scanners seen in the UCSD network telescope and in ETH Zurich is very similar with the exception of China. In our dataset China is a significant source of SIP scanning accounting for 9.90% of the total scanners population. On the UCSD dataset China is ranked 27th. More importantly, the geographical distribution of exploiters is particularly interesting, since it is dominated by Western countries and United States in particular, which is the most strongly represented country with 27.11% of the exploiters. In contrast, the geographical distribution of scanners is dominated by Eastern countries. US is not sourcing sipscanning, which is remarkable since the analysis of the botnet has shown a strong presence in the United States [27]. This observation shows that information about scan repliers was communicated from scanning to attacking bots through unknown channels.

Next, we examine the exploitation activity on port numbers related to SIP. Figure 5 shows the number of exploitation alerts targeting sipscan repliers on ports 5060, 5061, 5070 and 80. Ports 5060, 5061 and 5070 are used by SIP for control and data traffic. Moreover, the sipscan binary attempts to open a connection and gain administration privileges on port 80, where an HTTP server may provide remote administration

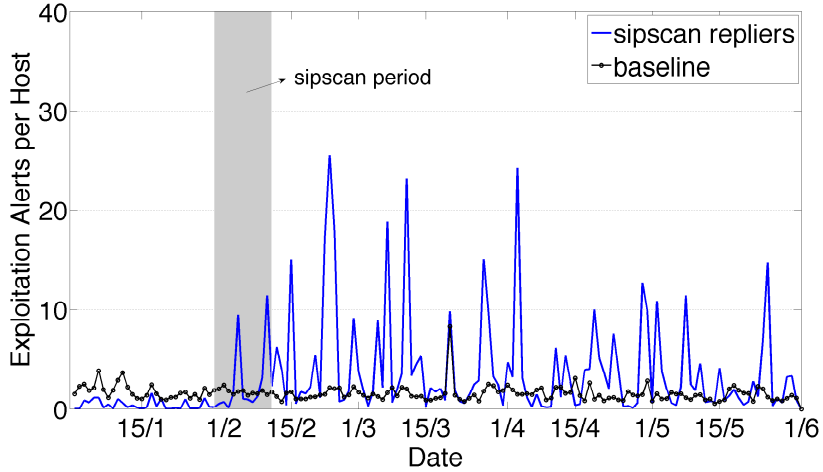


Fig. 2: Daily number of inbound exploitation alerts per target host over a period of 5 months. The two lines mark hosts that replied and that did not reply (baseline) to the sipscan. The shaded region marks the duration of the sipscan.

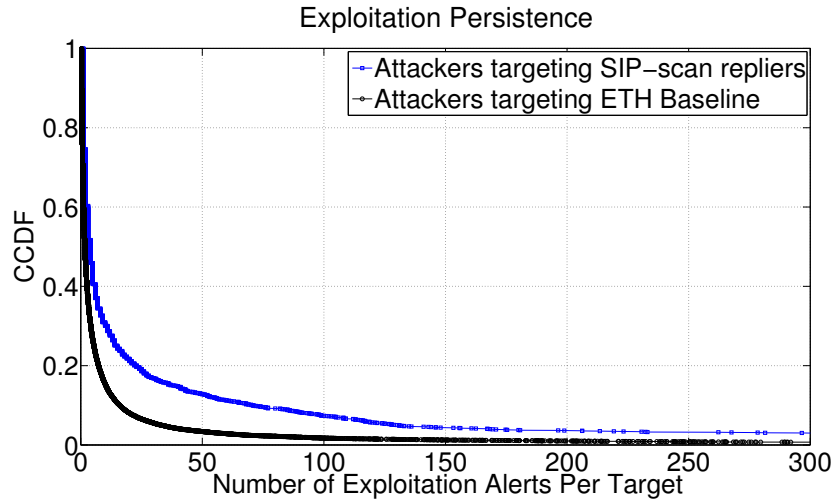


Fig. 3: Persistence of exploitation attackers for sipscan repliers and ETH-Baseline.

TABLE I: Top 10 countries used by the sipscanners compared to the respective countries for exploitation attack originators. Geolocation data for sipscan sources and exploitation attack originators was obtained using the MaxMind GeoIP Lite Database[26].

<i>sipscanners</i>		<i>CAIDA</i>	<i>sipscanners</i>		<i>ETH</i>	<i>Exploiters</i>		<i>ETH</i>
Rank	%	Country	%	Country	%	Country		Country
1	12.55	Turkey	10.06	Indonesia	27.11	United States		
2	12.54	India	9.72	Turkey	12.70	Canada		
3	8.64	Brazil	7.32	China	9.90	China		
4	7.23	Egypt	6.86	Brasil	7.01	Switzerland		
5	5.77	Indonesia	6.52	Egypt	4.98	Germany		
6	5.59	Romania	5.94	India	4.78	Taiwan		
7	5.58	Russian Federation	4.80	Thailand	4.31	Japan		
8	5.36	Vietnam	4.06	Philippines	3.31	India		
9	5.10	Thailand	3.71	Russian Federation	2.95	Russian Federation		
10	3.01	Ukraine	3.20	Romania	2.88	Brazil		

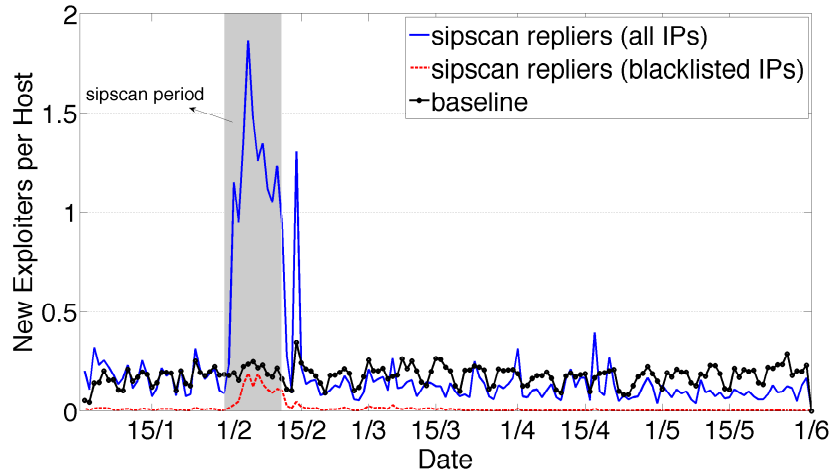


Fig. 4: Daily number of new offending IP addresses per target host for sipscan repliers and the baseline.

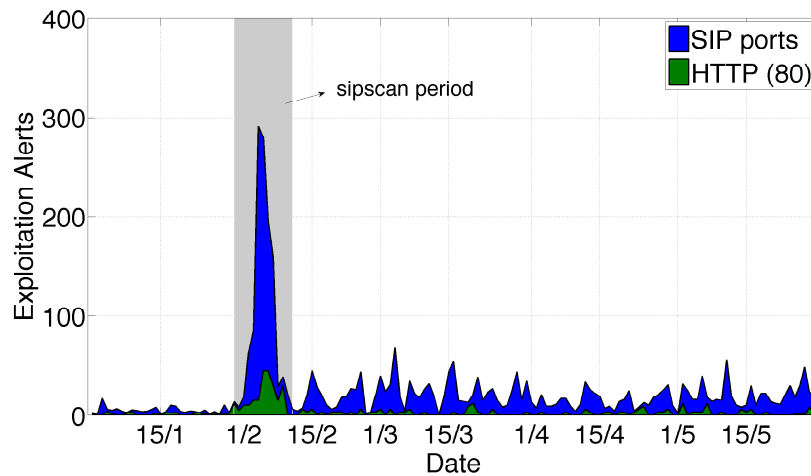


Fig. 5: Alert volume for exploitation attempts targeting SIP related ports.

to SIP servers [28]. Figure 5 shows a sharp increase of exploitation activity targeting SIP ports during and after the sipscan. Before, the sipscan we observe on a daily basis less than 12 exploitation alerts targeting SIP ports and 3 alerts targeting port 80. During the sipscan period, these numbers jump to 135 and 27, respectively, exhibiting approximately a ten-fold increase. Moreover, during the sipscan period 22% of all inbound exploitation alerts are on SIP ports. In the post-scan period we observe that these values drop, but still remain significant compared to the pre-sipscan period. Specifically, the daily number of exploitation alerts targeting SIP ports and port 80 are 5 and 21, respectively.

Finally, we inspect whether the inbound exploitation attacks against SIP ports occur in closer temporal proximity to the sipscan than attacks targeting non-SIP ports. In Figure 6 we compute the time between the first scan flow and the first exploitation alert towards a victim. We differentiate between exploitation alerts that target SIP ports and all other ports. In the former case, we see that in 98% of the incidents the attack

occurs within one week after the sipscan event, whereas the median is 2.7 days. In the latter case 95% of the incidents occur within one month after the sipscan event, whereas the median is 4.2 days.

To summarize the key findings of this section, we first observe a steep increase in exploitation alerts against sipscan repliers right after the sipscan, which is associated only with sipscan repliers and not with other hosts in the monitored infrastructure. Second, we observe that the attackers associated with the increase appear for the first time during the sipscan and were not active before. Third, we observe a sharp increase in exploitation alerts towards SIP ports and show that these exploitation attempts happen in close temporal proximity to the sipscan. We believe these findings constitute sufficient evidence that the sipscan was the precursor of a subsequent large-scale exploitation activity targeting sipscan repliers.

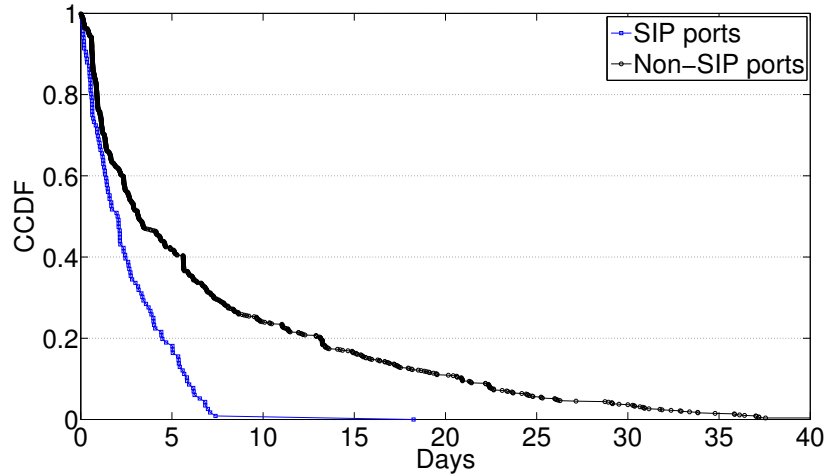


Fig. 6: CCDF of time between scanning and inbound exploitation attempt of a host for alerts on SIP and all other ports.

B. Sality alert classification and outbound exploitation activity

In Sections IV and V, we analyzed the inbound scanning and exploitation activity towards the monitored network. In this section, we shift our attention to the outbound IDS alerts generated by sipscan repliers and analyze the new behavioral patterns that emerge.

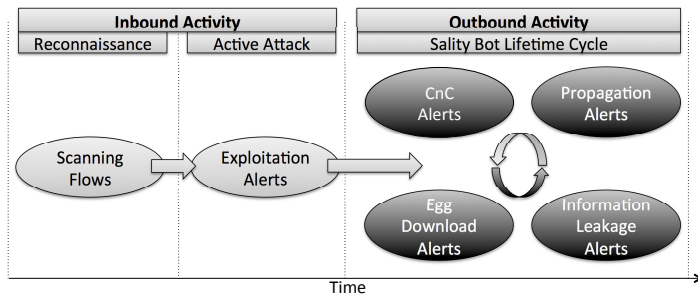


Fig. 7: Sality bot lifecycle

Sality is a very sophisticated and resilient malware. It consists of three main components: an infector, a communicator, and a downloader. The infector employs advanced propagation techniques to spread by infecting executable files and replicating on removable drives and network shares. The malware is polymorphic and encrypted, which makes its detection and remediation using traditional signature-based anti-viruses very challenging. The communicator uses a C&C architecture to pull instructions from the botmaster and to push harvested confidential data from the victim. Older versions of Sality utilized a centralized HTTP-based C&C channel, whereas recent versions form an unstructured P2P network, where each peer can be used to disseminate instructions within the botnet and to receive and forward information from other bots. The communicator is responsible for fetching a list of URLs hosting malware, which are subsequently fed to the downloader. In this way, the Sality botnet can be leveraged to push additional malware to the infected population, operating as a pay-per-install infrastructure [29].

Figure 7 illustrates the different stages that a Sality bot

undergoes during its lifetime. The first two stages correspond to the enumeration and active exploitation of the victim, which occur during the pre-infection phase. After the victim has been compromised, the Sality bot will typically undergo through a cycle of recurring actions. First, it will frequently contact its C&C servers to receive instructions and update its list of bootstrap nodes. Second, it will attempt to fetch malicious eggs to either update itself or to install additional malware on the victim. Third, it will try to leak sensitive information harvested from the victim, either by directly sending this information to a C&C server or by exploiting popular services and social networks, such as Dropbox and Facebook, to exfiltrate the data. Finally, it will attempt to propagate to vulnerable hosts by exploiting vulnerabilities related to the remote desktop and network shares services.

In Table II, we list the Snort identifiers (SIDs) and their official short description for relevant signatures that are triggered in our data. To compile the list, we manually analyzed the outbound alerts generated by sipscan repliers. We found the new types of alerts that emerged in the post-scan period and inspected their signatures in order to identify specific behaviors. We group signatures into four categories shown in Table II.

Signatures in the group *C&C Communication* detect the activity triggered by a bot when calling its controller for instructions. In the case of the HTTP version of the Sality bot, the signatures in the SID range (2404138:2404156) are triggered when a set of known blacklisted C&C servers are contacted, whereas the signature (2000348) detects the setup of an IRC channel, which is used by the bot and the controller to communicate. The remaining alerts are related to the P2P version of the bot and are triggered when the bot is either attempting to join the P2P network, instantiating a new P2P connection, or fetching the latest peers list.

Signatures in the group *Exfiltration* are tailored to detect the exfiltration of confidential data. The SIDs (5,2006380) are triggered when passwords or email addresses are sent from the intranet unencrypted. The signature (2010784) is triggered when the bot is attempting to leak sensitive information using

TABLE II: Snort signatures related to Sality bot lifecycle.

SID	Signature Description
[C&C Communication] Communication with botnet controller.	
2404138:2404156	ET DROP Known Bot C&C Server Traffic TCP/UDP
2000348	ET ATTACK_RESPONSE IRC - Channel JOIN on non-std port
2000334	ET P2P BitTorrent peer sync
2009971	ET P2P eMule KAD Network Hello Request
2008581	ET P2p BitTorrent DHT ping Outbound
2010142	ET P2P Vuze BT UDP Connection Outbound
2008584	ET P2P BitTorrent DHT announce_peers request
2181	P2P BitTorrent transfer
[Exfiltration] Possible leakage of sensitive user data.	
5	SENSITIVE-DATA Email Addresses Outbound
2006380	ET Policy Outgoing Basic Auth Base64 HTTP Password detected unencrypted
2010784	ET CHAT Facebook Chat POST Outbound
2000347	ET ATTACK_RESPONSE IRC - Private message on non-std port
1463	CHAT IRC message Outbound
[Propagation] Attempted infection of vulnerable hosts.	
2007695,2008070	ET User-Agent Malware overflow attempt
4060	POLICY RDP attempted administrator connection request
2006546	ET SCAN LibSSH Based SSH Connection - BruteForce Attack
2002383	ET SCAN Potential FTP Brute-Force attempt
3817	TFTP GET transfer mode overflow attempt
2010643	ET SCAN Multiple FTP Administrator Login Attempts- Brute Force Attempt
2001972	ET SCAN Behavioral Unusually fast Terminal Server Traffic, Potential Scan or Infection
2001569	ET SCAN Behavioral Unusual Port 445 traffic
[Egg Download] Possible download of malicious executable.	
2009897	ET MALWARE Possible Windows Executable sent when remote host claims to send a Text File
19270	POLICY attempted download of a PDF with embedded Javascript
15306	WEB-CLIENT Portable Executable binary file transfer
2003546	ET USER Agents Suspicious User agent Downloader
2007577	ET TROJAN General Downloader Checkin URL
2012648	ET Policy Dropbox Client Downloading Executable
2009301	ET Policy Megaupload file download service access

Facebook’s POST mechanism. This alert should be expected to generate a significant amount of false positives, since it is also triggered when a user sends a legitimate Facebook message. However, a sudden sharp increase in the amount of Facebook POST operations could signify a malicious activity. The signatures with SIDs (2000347,1463) are triggered when information is exfiltrated using an IRC channel.

Signatures in the group *Propagation* are generated when the bot is attempting to infect exposed vulnerable hosts. The main targeted vulnerabilities are the MS-LSASS buffer overflow and the MS-WebDav vulnerability related to services used for accessing remote network shares. The set of signatures shown in Table II are fine-tuned to detect brute force privilege escalation attacks (4060,2006546,2002383,2010643), buffer overflow exploitation attempts (2007695, 2008070,3817), and targeted scanning on these services (2001972 ,2001569).

Finally, signatures in the group *Egg Download* correspond to attempts made by the bot to fetch a malicious binary from a remote domain. The downloaded executable can be either an update of Sality’s own code or can correspond to a new malware pushed to the infected population. Signatures with SIDs (15306,2003546,2007577) detect the activity of Sality’s downloader module when attempting to check a suspicious URL or when a binary download is initiated. Sality tries to

obfuscate the downloaded binary by hiding it in seemingly legitimate files, such as Text and PDF documents. This activity is detected by signatures with SIDs (2009897,19270). The obfuscation is used to evade detection by cloud providers, such as Dropbox and Megaupload, which are exploited in order to host the malicious content. Signatures with SIDs (2012648,2009301) detect the download of executables from these sites.

Figure 8 shows the average number of C&C alerts triggered by sipscan repliers and baseline hosts. For sipscan repliers, we differentiate between IRC and P2P C&C alerts, whereas for the baseline we include both types of alerts. After the sipscan, we see a sharp increase in the IRC C&C alerts, which indicates that hosts are attempting to contact known malicious IRC servers operating as controllers. This behavior continues for approximately two months, during which we see daily on average 2.4 C&C alerts per sipscan replier. However, on April 11 (day 111) there is a clear shift in the pattern of triggered signatures: the volume of IRC alerts suddenly drops, while the volume of P2P alerts rises. This signifies a likely upgrade in the mode of C&C communication of the Sality botnet.

Figure 9 illustrates the daily number of *Egg Download* alerts per sipscan replier and baseline host. After the sipscan, we observe 4 malware downloading spikes, during which the

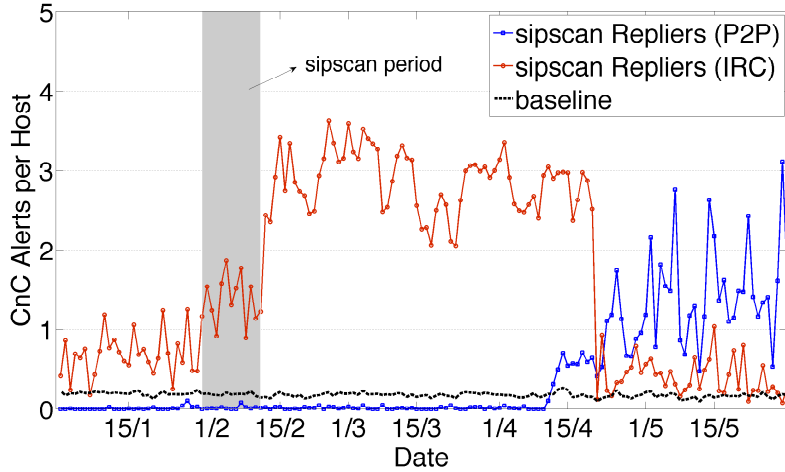


Fig. 8: Daily number of outbound C&C alerts per host for sipscan repliers and for baseline hosts over a period of 5 months. We show IRC and P2P C&C alerts in different lines.

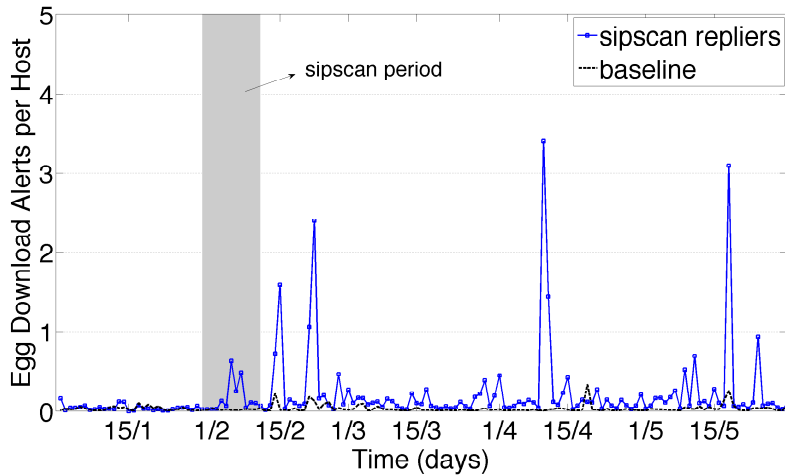


Fig. 9: Daily number of outbound Egg Download alerts per host for sipscan repliers and for baseline hosts over a period of 5 months.

daily alert count ranges from 1.6 to 3.4 per sipscan replier. The spike that occurs on April 11 (day 111), seems to be associated with the shift in the communication method used to contact the controller shown in Figure 8. We believe that during that event the Sality botnet pushed a major update to the infected population, upgrading itself from the centralized HTTP to the fully decentralized P2P version.

In Figure 10, we show the daily number of *Propagation* alerts per local host for sipscan repliers and baseline hosts. We see that after the sipscan the number of outbound exploitation attempts originating from the sipscan repliers increases drastically, exhibiting an average daily value of 1.2 alerts per host compared to only 0.21 alerts per baseline host. The most dominant alerts of this group are the privilege escalation attempts with SIDs (4060,2006546,2002383,2010643) accounting for 72% of the observed activity.

Finally, Figure 11 illustrates the daily number of infor-

mation leakage alerts per local host for sipscan repliers and baseline hosts. Again we see a sharp increase in the number of exfiltration alerts for sipscan repliers in the post-sipscan period, where the daily average increases from 4.7 to 18.2 alerts per host. The triggered alerts are dominated by the signature *ET CHAT Facebook Chat POST Outbound*, which accounts for 83% of all alerts. However, this signature is also triggered by legitimate user activity and may introduce a significant number of false positives. This is reflected in the high baseline and in the pre-sipscan period, when it accounts on average for 4.7 alerts per host. Although the baseline for this alert group is high, we can still see a clear increase in the post-sipscan period when its alert volume quadruples.

C. Sality-bot infections

In Section V-B we discovered major changes in the alert patterns of sipscan repliers that correlate with the behavior of

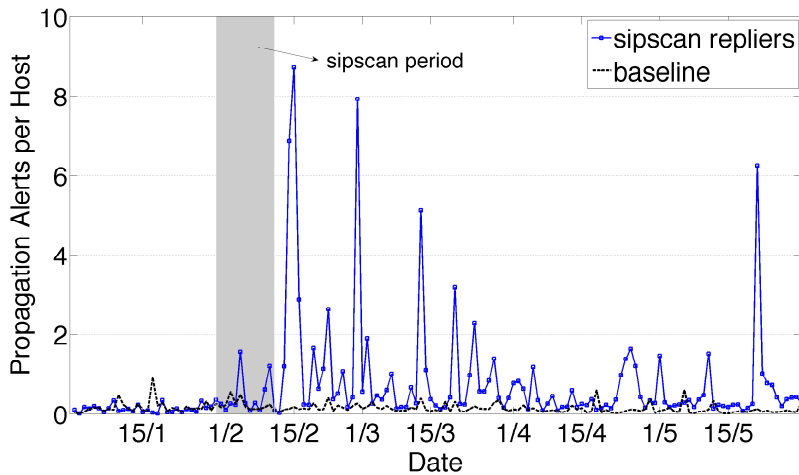


Fig. 10: Daily number of outbound Propagation alerts per host for sipscan repliers and for baseline hosts over a period of 5 months.

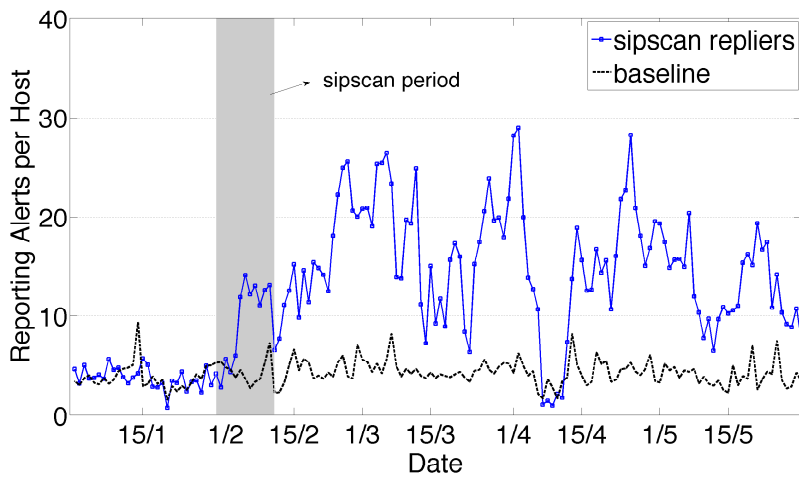


Fig. 11: Daily number of outbound Exfiltration alerts per host for sipscan repliers and for baseline hosts over a period of 5 months.

the Sality bot. In this section, we build a heuristic to identify this behavioral shift and extract likely Sality infections. Our goal is not to build a general purpose detector, but rather a systematic way to identify infected sipscan repliers in the monitored network. We use our heuristic to conservatively estimate a lower bound on the success rate of the sipscan in terms of infected hosts.

Our heuristic is summarized in Algorithm 1. We focus on sipscan repliers that were subsequently attacked. Then we find repliers that exhibit a persistent increase in outbound exploitation activity for the four signature classes listed in Table II, while their respective activity in the pre-sipscan period is low. In particular, for the four classes in Table II, we first compute the number of alerts per day each internal host generates. Our heuristic then finds and keeps hosts that trigger in the pre-sipscan period fewer alerts per day than the corresponding baseline of that day plus a tolerance of $1.5 \times$ the inter-quartile

range of the baseline. If a host has more alerts per day even for a single day, then it is discarded from further consideration because it is either already infected or it generates a large number of false positives. Second, our heuristic makes the same comparison in the post-sipscan period. If the daily alert count is consistently above the tolerance threshold, then it constitutes an indication of compromise activity. To assess whether this increase persists, we count the number of daily bins where it is above the threshold and tolerate only 5% of the post-sipscan bins where this condition is not met. We consider only the bins in which a host has generated at least one alert of any type.

Our heuristic takes a conservative approach by introducing several conditions to make a Sality infection assessment. It is possible, however, that a Sality bot exhibits some of the post-sipscan behaviors illustrated in Figure 7 but not all. For example, some examined hosts show persistent signs of

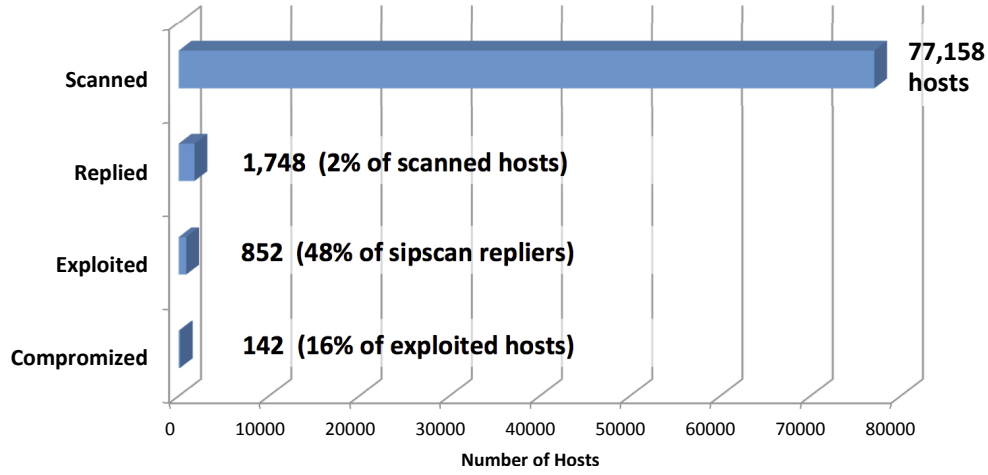


Fig. 12: Sality sipscan ETH turnover

Input:

B_T^S : mean count of S type alerts generated by Baseline hosts on day T

I_T^S : IQR of S type alerts generated by Baseline hosts on day T

R_T^S : mean count of S type alerts generated by sipskan repliers on day T

$S = \{\text{CnC Communication, Reporting, Propagation, Egg Download}\}$

Result: Returns *true* if the examined host is infected, *false* otherwise.

foreach alert type S **do**

$BelowThreshCount = 0$;

for $T_i = 1:T_{max}$ **do**

if $isHostActiveAt(T_i)$ eq *false* **then** *next*;

$SignificanceThresh = B_{T_i}^S + 1.5 * I_{T_i}^S$;

if $T_i \leq T_{scan}$ **then**

if $R_{T_i}^S > SignificanceThresh$ **then**
 | return *false*;

end

else

if $R_{T_i}^S \leq SignificanceThresh$ **then**
 | $BelowThreshCount += 1$;

end

end

if $BelowThreshCount / (T_{max} - T_{scan}) > 0.05$ **then**
 | return *false*;

end

end

end

return *true*;

Algorithm 1: Pseudo-code for identifying Sality-bot infections

C&C communication and attempts to propagate, but do not attempt to leak data. Others attempt to exfiltrate data, but do not frequently visit malicious domains to fetch malware. By tailoring our heuristic to only make an assessment if all alert types in the post-sipscan period exhibit a persistent increase, we attempt to minimize possible false positives even if we introduce a number of false negatives. This way, we believe we can estimate a lower bound of the Sality infections that occurred in our infrastructure.

Our heuristic identified a total of 142 Sality infections in our IDS dataset. Figure 12 summarizes the estimated success rate of the Sality botnet in the monitored network. In the first stage of reconnaissance, 77,158 exposed ETH Zurich IPs were scanned. Out of these only 1,748 (2%) hosts replied to the scanners. Almost half of the sipskan repliers, specifically 48%, were subsequently the targets of inbound exploitation attacks. Based on our heuristic we identified that 142 hosts showed persistent signs of infection during the post-sipscan period. Therefore, the sipscan turnover, i.e. the percentage of hosts that were infected out of the sipskan repliers, was 8%.

VI. DISCUSSION

The quality of IDS alerts we study in Section V heavily relies on the accuracy of the inferences made by the Snort sensor deployed in our infrastructure. Snort has been criticized for generating an excessive number of false positives, often exceeding 99% [30], [31]. Such high false positive rates can introduce significant bias in our measurements, resulting in skewed results. However, in this work we have focused on signatures which, based on our previous work [32], [33], were shown to be reliable, generating only a small number of false positives. Specifically, in [32] we performed a thorough evaluation of the alerts being triggered by Snort in our infrastructure and identified signatures that generate a large number of false positives. These alerts have been excluded from the current work. Moreover, in [33] we introduced a complexity criterion to evaluate the effectiveness of a Snort signature in terms of correctly identifying malicious activity. The alerts analyzed in Section V are triggered by highly complex signatures, which our analysis in [33] has shown to be more reliable, generating a low number of false positives.

VII. CONCLUSIONS

In this work, we provide new insights about Internet scanning, focusing on an interesting Internet-wide scanning event orchestrated by a large botnet. Using a unique dataset of both unsampled Netflow records and IDS alerts collected from a large academic network, we assess the effectiveness of scanning in terms of targeted hosts that replied and hosts that were eventually compromised. We find that 2% of the scanned hosts replied to the scanners and at least 8% of the repliers were eventually compromised by subsequent exploitation attempts. In addition, our work provides new insights about the “/0” sipscan orchestrated from the Sality botnet [5]. We find the sipscan was only a pre-cursor of subsequent exploitation attacks towards sipscan repliers. The attack escalated, leading to at least 142 infected hosts in the monitored network. Furthermore, we observe a segregation of roles between scanners and exploiters, which originate from different geographical locations. Finally, we observe that the sipscan originated from 352,350 additional IP addresses. Our study provides a novel view into Internet scanning from the side of scan repliers.

REFERENCES

- [1] S. Shin, R. Lin, and G. Gu, “Cross-analysis of botnet victims: New insights and implications.” in *RAID*, ser. Lecture Notes in Computer Science, R. Sommer, D. Balzarotti, and G. Maier, Eds., vol. 6961. Springer, 2011, pp. 242–261. [Online]. Available: <http://dblp.uni-trier.de/db/conf/raid/raid2011.html#ShinLG11>
- [2] M. Bailey, E. Cooke, F. Jahanian, Y. Xu, and M. Karir, “A survey of botnet technology and defenses.”
- [3] E. Glatz and X. Dimitropoulos, “Classifying internet one-way traffic,” in *Proc. of the 2012 ACM conf. on Internet measurement*. NY, USA: ACM, 2012.
- [4] Z. Durumeric, E. Wustrow, and J. A. Halderman, “ZMap: Fast Internet-wide scanning and its security applications,” in *Proceedings of the 22nd USENIX Security Symposium*, Aug. 2013.
- [5] A. Dainotti, A. King, K. Claffy, F. Papale, and A. Pescap, “Analysis of a “/0” Stealth Scan from a Botnet,” in *Proceedings of the 2012 ACM conference on Internet measurement conference*, Nov 2012, pp. 1–14.
- [6] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, “Sip: Session initiation protocol,” U.S., 2002.
- [7] E. Cooke, F. Jahanian, and D. Mcpherson, “The zombie roundup: Understanding, detecting, and disrupting botnets,” 2005, pp. 39–44.
- [8] P. Barford and V. Yegneswaran, “An inside look at botnets,” in *Malware Detection*, ser. Advances in Information Security, M. Christodorescu, S. Jha, D. Maughan, D. Song, and C. Wang, Eds. Springer, 2007, vol. 27, pp. 171–191.
- [9] C. Rossow, D. Andriess, T. Werner, B. Stone-Gross, D. Plohmann, C. J. Dietrich, and H. Bos, “Sok: P2pwned - modeling and evaluating the resilience of peer-to-peer botnets,” in *IEEE Symposium on Security and Privacy*, 2013, pp. 97–111.
- [10] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freiling, “Measurements and mitigation of peer-to-peer-based botnets: A case study on storm worm,” in *Proceedings of the 1st Usenix LEET Workshop*, Berkeley, CA, USA, 2008, pp. 9:1–9:9.
- [11] M. Abu Rajab, J. Zarfoss, F. Monrose, and A. Terzis, “A multifaceted approach to understanding the botnet phenomenon,” in *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC ’06, NY, USA, 2006.
- [12] B. Stone-gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, “Your botnet is my botnet: Analysis of a botnet takeover.”
- [13] F. C. Freiling, T. Holz, and G. Wicherski, *Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks*, 2005.
- [14] A. Karasaridis, B. Rexroad, and D. Hoeflin, “Wide-scale Botnet Detection and Characterization.”
- [15] G. Gu, R. Perdisci, J. Zhang, and W. Lee, “BotMiner: Clustering Analysis of Network Traffic for Protocol and Structure-Independent Botnet Detection,” in *USENIX Security Symposium*, 2008, pp. 139–154.
- [16] G. Gu, J. Zhang, and W. Lee, “BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic,” in *NSDI*, 2008.
- [17] P. Bacher, T. Holz, M. Kotter, and G. Wicherski, “Know your enemy: Tracking botnets,” <http://www.honeynet.org/papers/bots>, 2008.
- [18] V. Yegneswaran, P. Barford, and V. Paxson, “Using honeynets for internet situational awareness,” in *Fourth Workshop on Hot Topics in Networks (HotNets IV)*, 2005.
- [19] Z. Li, A. Goyal, and Y. Chen, “Honeynet-based botnet scan traffic analysis,” in *Botnet Detection*, ser. Advances in Information Security, 2008, vol. 36, pp. 25–44.
- [20] Z. Li, A. Goyal, Y. Chen, and V. Paxson, “Towards situational awareness of large-scale botnet probing events,” *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 1, pp. 175–188, 2011.
- [21] SWITCH, “Swiss Tele Communication System for Higher Education,” <http://www.switch.ch/>.
- [22] “Anonymous postmasters early warning system,” <http://www.apews.org/>, [Online; accessed 13-03-2014].
- [23] “Shadowserver foundation,” <https://www.shadowserver.org/>, 2014, [Online; accessed 13-03-2014].
- [24] “Dshield: Internet storm center,” <http://www.dshield.org/>, 2014, [Online; accessed 13-03-2014].
- [25] “Threatexpert - automated threat analysis,” <http://www.threatexpert.com/>, 2014, [Online; accessed 13-03-2014].
- [26] M. Lite, “GeoIP Database,” <http://dev.maxmind.com/geoip/legacy/geolite/>, 2014, [Online; accessed 03-03-2014].
- [27] Falliere, “Sality: Story of a peer-to-peer viral network,” 2011.
- [28] N. Falliere, “A distributed cracker for voip,” <http://www.symantec.com/connect/blogs/distributed-cracker-voip>, 2011.
- [29] J. Caballero, C. Grier, C. Kreibich, and V. Paxson, “Measuring pay-per-install: The commoditization of malware distribution,” in *Proceedings of the 20th USENIX Conference on Security*, ser. SEC’11, Berkeley, CA, USA, 2011, pp. 13–13.
- [30] K. Julisch and M. Dacier, “Mining intrusion detection alarms for actionable knowledge,” in *In The 8th ACM International Conference on Knowledge Discovery and Data Mining*, 2002, pp. 366–375.
- [31] C. Kruegel and W. Robertson, “Alert verification - determining the success of intrusion attempts,” in *Proc. First Workshop the Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA 2004)*, 2004, pp. 1–14.
- [32] E. Raftopoulos and X. Dimitropoulos, “Detecting, validating and characterizing computer infections in the wild,” in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. ACM, 2011, pp. 29–44.
- [33] —, “A quality metric for ids signatures: In the wild the size matters,” *EURASIP Journal on Information Security*, Dec 2013.