Tracking Malicious Hosts on a 10Gbps Backbone Link*

Magnus Almgren and Wolfgang John

Computer Science and Engineering
Chalmers University of Technology, Sweden
{first.lastname}@chalmers.se

Abstract. We use anonymized flow data collected from a 10Gbps backbone link to discover and analyze malicious flow patterns. Even though such data may be rather difficult to interpret, we show how to *bootstrap* our analysis with a *set of malicious hosts* to discover more obscure patterns. Our analysis spans from simple attribute aggregates (such as top IP and port numbers) to advanced *temporal analysis of communication patterns* between normal and malicious hosts. For example, we found some complex communication patterns that possibly lasted for over a week. Furthermore, several malicious hosts were active over the whole data collection period, despite being blacklisted. We also discuss the problems of working with anonymized data. Given that this type of privacy-sensitive backbone data would not be available for analysis without proper anonymization, we show that it can still offer many novel insights, valuable for both network researchers and practitioners.

Keywords: Network Security; Malicious Traffic; Internet Backbone.

1 Introduction

The amount of Internet malware in circulation has increased and is forecasted to increase even further [1]. Also the types of attacks have changed over time and are today very different from the ones seen a decade ago. From being a way to gain esoteric prestige, the attacks nowadays are connected to organized crime [2]. It is important to understand how prevalent malicious code is, how it spreads, how many "normal" users are infected, and what happens when one is infected.

There are several orthogonal methods to find partial answers to these questions. For example, companies or other large organizations can analyze the traffic in their networks. In these settings, especially given the fact that the organization has a budget for security incident investigation, there often exists a security policy with enforcement. That is, as certain security mechanisms are used the data from such organizations will only show a subset of possible security incidents.

Antivirus companies [3,4], with their software ubiquitously deployed on many computers around the world, can also collect certain data from their customers to analyze larger trends. However, some information is sensitive to export from the client and the data are again skewed; they come from computers where security mechanisms have been installed and where the owners (presumably) are security conscious.

* This work is supported by the Swedish Civil Contingencies Agency (MSB) and SUNET. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 257007.

Large networks of honeypots [5,6] offer invaluable insights into malware behavior but again the data are biased as these are not regular computers with regular users. Given their specialized nature, it is also expected that other hosts on the network have a higher-than-average security protection; the administrators that spend time to install and monitor a honeypot usually also invest time into other (simpler) security mechanisms. Similar reasoning goes for DShield/SANS' aggregated data [7,8].

In our work, we analyze traffic from the backbone of the Swedish University Network SUNET. This is a high-level cross-section of traffic from a very large domain, giving us an aggregated view of a very large number of hosts containing both security-conscious users (i.e., researchers in security) as well as less sophisticated users (students using the computer as a means to an end). By analyzing such traffic, we hope to gain a different, and potentially more general, view of current malware behavior than the approaches described above. However, we acknowledge that our cross section of users is also skewed, albeit in a different way than above. SUNET mainly provides high-speed Internet access to academic institutions in Sweden, meaning that a majority of the users are either students or other people connected to academic institutions or research environments, but there are also other types of users such as museums and some government agencies.

Our analysis of malicious behavior and the corresponding collection is focused on *anonymized flows* (i.e. summaries of packet streams between communication endpoints) and not full packet payload. The advantages include a more manageable amount of data that are less privacy-invasive than a full packet payload capture. The disadvantages include no ground truth and a limited ability to further refine or validate our results. Even though anonymized flow data may stymie some type of analysis, we show that such data can still be used to discover typical *malicious flow patterns* that we then investigate in detail. We consider our results here as a survey with possibilities of future extensions, both when it comes to the extent of the data analyzed and the methods used.

The rest of the paper is organized as follows. In Section 2 we describe the collection of traffic and explain the type of data available for analysis. In Section 3, we describe the general characteristics of this data. We then outline the problem of finding malicious behavior in Section 4 and formally describe the assumptions and requirements we need for the analysis of malicious flows. In Section 5, we analyze the behavior of malicious hosts. In Section 6, we describe related work. The paper is concluded in Section 7.

2 Description of the Data Collection

2.1 Measurement Setup

We collected backbone traffic on an OC-192 (10Gbps) link in the core-backbone of SUNET, the Swedish University Network. Its current version, *OptoSUNET*, is a star structure over leased fiber, with a central exchange point in Stockholm. OptoSUNET connects all SUNET customers redundantly to a core network in Stockholm, as depicted in Figure 1. Traffic routed to the international commodity Internet is carried on three links between SUNET and NORDUnet, where NORDUnet peers with Tier-1 backbone providers, large CDNs (Content Distribution Networks) and other academic networks. We used an existing 10Gbps measurement infrastructure [9] to collect traffic on one of the 10Gbps links between SUNET and NORDUnet, indicated in black color in Figure 1.

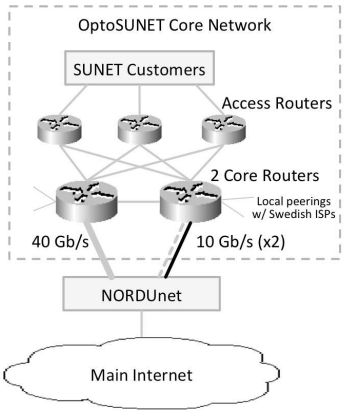

Fig. 1: OptoSUNET core topology. All SUNET customers are via access routers connected to two core routers. The SUNET core routers have local peering with Swedish ISPs, and are connected to the international commodity Internet via NORDUnet. SUNET is connected to NORDUnet via three links: a 40Gbps link and two 10Gbps links. Our measurement equipment collects data on the first of the two 10Gbps links (black) between SUNET and NORDUnet.

Our measurement hardware includes two measurement nodes on site and one additional processing platform at our university. At the core network in Stockholm, we apply optical splitters to tap the two OC-192 links, one for each direction. The splitters are attached to the measurement nodes on-site, which also preprocess the traces, including prefix-preserving IP address anonymization [10]. We always collected network data simultaneously for both directions. For the final analysis, we transferred anonymized network flows to the processing platform at Chalmers University.

2.2 Description of the Collected Data: Unidirectional Flows

We ran our data collection every week for 24 hours with *crl_flow* of the CoralReef Suite [11]. We define flows by the unidirectional sequence of packets sharing a 5-tuple of {`sourceIP`,`destinationIP`,`sPort`,`dPort`,`proto`}. Flows are then further discriminated by a 5-minute timeout interval, i.e., two packets sharing the same tuple belong to the same flow if their timestamps are within the given interval. A sample flow summary of crl_flow with anonymized IP addresses is shown in Table 1. Flow summaries include the identifying 5-tuple, where the *proto* represents transport protocol numbers as assigned by IANA, such as 6 for TCP, 17 for UDP and 1 for ICMP. Note that in the case of ICMP the port fields contain the type and code, respectively. Other meaningful fields are *pkts* and *bytes*, containing the number of packets and bytes seen within the flow interval; and *firstTS* and *latestTS*, representing POSIX timestamps of the first and last captured packet in a flow, respectively. Note that we do not normally see any TCP/IP header information apart from the ports and timestamps described above.

Table 1: Two lines describing the flow output from CoralReef (IP addresses anonymized).

sourceIP	**destinationIP**	**proto**	**ok**	**sport**	**dport**	**pkts**	**bytes**	**flows**	**firstTS**	**latestTS**
192.168.52.11	74.125.43.147	6	1	445	3995	3	120	1	t_0^1	t_n^1
192.168.10.69	74.125.43.101	1	1	3	1	1	56	1	t_0^2	t_n^2

2.3 Measurement Bias and Errors

According to SNMP statistics [12], the applied load-balancing mechanisms by SUNET assigned about 30% of all inbound but only 15% of the outbound traffic volume to

the 10Gbps link measured, and the rest to the alternative links. This type of sampling bias is hard to quantify, since the routing policies are outside our control and differ for incoming and outgoing traffic. They may also change without our knowledge.

The routing of the outgoing traffic is decided by the organization the traffic is originating from, meaning that different rules govern different parts of the "inside" network. We have observed that we are blind to outgoing traffic from some IPs, for some hosts we only see a subset of their traffic and for yet others, we may detect all traffic.

The policy for incoming traffic is slightly more uniform, but still complex due to the setup of different peering points and agreements, introducing e.g. hot potato routing effects. In general, we see a subset of all traffic, depending on routing decisions based on a three-tuple of {`sourceIP, sPort, destinationIP`} for TCP/UDP flows. Traffic from some peering points are not visible at all at our measurement point.

There are also a few caveats of the experimental hardware setup. Even though we normally had no traffic loss within collection periods, there were two exceptions. Firstly, the measurement cards can sometimes loose synchronization with the OC-192 PoS framing, so we proactively restarted the collection in 3h periods, leading to missing packets in the second between such data collection periods. Secondly, there were four short, but immense traffic surges, where traffic was increasing from the normal rate of $<200k$ to $>400k$ packets per second. During these surges, our nodes could not keep up with the speed and dropped packets, which was logged by the measurement cards.

Finally, the measurements were done over an operational large network, meaning that parameters change over the course of the data collection. For example, on April 22, we saw a spike of traffic over the outgoing link we were monitoring, as one of the alternative routes was down for a short period of time. It is important to understand the limitations of the experiment setup for correct analysis of the data. We can reason about data we captured, but we need to be careful when interpreting missing data; a flow may be missing because it was never sent but it may also be missing because it was routed around our measuring point.

3 Overall Data Characteristics

In this section, we describe the overall data characteristics of the captured flows. Table 2 shows traffic statistics of the collection days used for this study.[1] The first observation is that we see many more incoming than outgoing flows, mainly due to the load-balancing mechanisms we explained in Section 2.3.

For the incoming traffic, the transport protocol breakdown in terms of flow numbers was about 42% TCP, 56% UDP, and 2% ICMP, while the outgoing link showed a slightly different protocol ratio with 28% TCP, 69% UDP, and 3% ICMP.[2] The number of flows changed over the data collection period but the traffic mix was relatively constant with two exceptions, April 8 and May 6. On these two days, we observed a larger number of flows due to major events involving a single host inside SUNET. This

[1] Note that our data include a substantial portion of incoming flows on UDP port 53, due to a RIPE DNS server located inside SUNET, serving over 400 zones. Traffic from and to port 53 on this server cannot be considered native SUNET traffic and we filtered it out for this study.

[2] Other protocols in the order of 0.1% are excluded. The values are often rounded to the nearest percent, and the sum is sometimes not exactly 100% (as in Table 2).

Table 2: A summary over the collection days and the corresponding traffic characteristics. The values in parenthesis in the columns for *Flows* and *Bytes* show the percentage of the traffic for TCP, UDP and ICMP respectively. Data for all days are captured in 2010 with a duration of 24h.

	Incoming Link			**Outgoing Link**		
Date	# Pkts / 10^9	# Flows / 10^8	# Bytes / 10^{12}	# Pkts / 10^9	# Flows / 10^8	# Bytes / 10^{12}
April 01	8.38	2.33 (39/59/2)	5.74 (83/17/0)	3.95	1.20 (27/70/3)	3.21 (58/41/0)
April 08	11.4	3.11 (48/50/2)	8.42 (85/15/0)	5.44	1.54 (27/70/3)	3.93 (52/47/0)
April 15	10.4	2.79 (40/58/2)	7.80 (84/16/0)	3.89	0.96 (28/69/3)	2.98 (54/45/0)
April 22	11.7	2.91 (41/57/2)	9.41 (87/12/0)	3.95	1.09 (29/69/2)	3.31 (61/38/0)
April 29	10.4	2.73 (41/58/2)	7.76 (86/13/0)	3.38	0.95 (30/68/2)	2.77 (57/43/0)
May 06	9.46	3.14 (46/52/2)	6.75 (84/16/0)	4.23	1.16 (30/67/2)	3.62 (58/41/0)

particular host was the target of a large number of connections from a widely scattered IP range via known IRC port numbers within short time periods. We see many incoming 1-pkt flows with 40 Bytes (probably RST packets), but also a substantial number of established connections involving exchange of small data portions. We suspect that this host was sending out *Botnet Command & Control* (C&C) traffic, where we see only the return traffic of the botnet zombies all over the world. Thus, we observe that malicious activity may even leave a footprint in large aggregates as the ones shown in Table 2.

Similarly, we can consider the protocol mix based on the number of bytes transferred. As can be seen, there was much more traffic sent over TCP than over UDP even though the number of flows of UDP exceeded the number of flows for TCP. Surprisingly, the UDP traffic accounted for as much as 43% of the outgoing traffic.

Table 3: Unique hosts during the data collection 2010-04-01.

	Inside SUNET	**Outside SUNET**
Incoming Link	Destination IPs 970,149	Source IPs 24,587,096
Outgoing Link	Source IPs 23,600	Destination IPs 18,780,894

In Table 3, we show how many unique IP addresses we saw on the links on the first collection day. Note that the destination address space for the incoming link is represented by the source address space on the outgoing link, due the opposing directions of the unidirectional links. Even though part of the difference between the address spaces observed can be explained by routing differences, there is a factor of 41 between the observed IPs inside SUNET between the two directions. We know from previous measurements that scanning operations, even though often unanswered from hosts inside SUNET, inflate the number of incoming destinations [13], and for that reason we have not done any closer analysis of such behavior on the data presented here.

Table 4: The number of unique source IP addresses found in the traffic on the *outgoing* link.

Date	April 1	April 8	April 15	April 22	April 29	May 6
Unique IP:s	23,600	26,398	12,223	76,143	12,218	12,603

In Table 4, we show the number of unique source IP addresses seen on the outgoing link. There are two artifacts we would like to highlight. First, on April 22 we see many

more source hosts. During a short period this day, one of the alternative routing links was down and more traffic was routed over the link we measure. By roughly excluding the 20 minutes the link was down, we have 16,823 unique sources, an *estimate* more similar in size to the other collection days.The other artifact is that onward from April 15 we see only about half of the sources, maybe because of a new routing policy. We briefly investigated how many of the sources on the outgoing link were also present in the data collected on the incoming link. Given the asymmetry of Table 3, one would expect a majority of the source IPs on the outgoing link also be present as destinations on the incoming link. In the data of April 1, it is 97.24%, confirming our expectations.

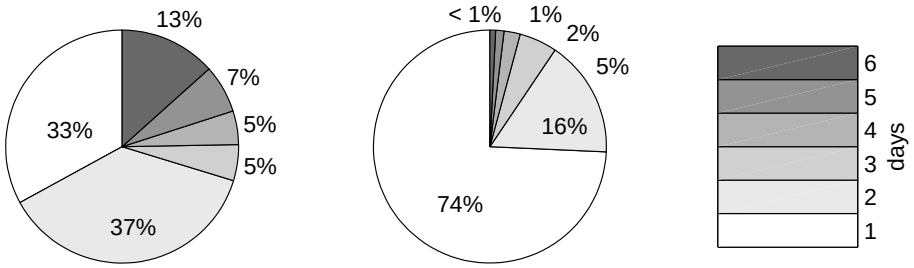

Fig. 2: The figure shows how many days a particular host is active. The pie chart to the left shows hosts inside SUNET (source IP addresses) while the pie chart to the right shows hosts outside SUNET (destination addresses). The figure is based on data collected on the *outgoing* link.

We also investigated how long we could detect traffic from a particular IP address. For example, an IP address may become unused with DHCP, a host may be removed from the network, or the routing policy is changed. In Figure 2, we have included two pie charts that show how many collection days a typical IP address was active. The chart on the left represents hosts inside SUNET while the one on the right represents hosts outside SUNET. Note we use the *estimate* for April 22, where the 20 minutes of exceptional traffic is excluded as described above.

Among hosts inside SUNET, we have found that a majority are only active on one or two data collection days even though 13% seem to be reoccurring every single data collection day. In our data, most hosts outside SUNET are also only visible a single day and there are very few hosts that reoccur over time.

4 Finding Malicious Hosts

We are interested in finding and analyzing typical *malicious flow patterns*. However, we face several problems when determining whether a host is malicious. First, its status can mostly only be determined in relation to a local security policy of allowed behavior. Transmission of data from a system, e.g., might be very permissible at some sites (research centers sharing results) but very suspicious at other sites (government agency analyzing pre-election numbers) where it could indicate exfiltration attacks. As we have

a bird's view over the network, we cannot make subjective judgment calls. However, certain behavior, such as spreading malware, is quite universally seen as malicious.

Second, as we only see flow information, it is difficult to verify our suspicions of a host's malicious status. Through statistical analysis of the anonymized flow data we can determine whether a host is behaving strangely compared to the mean, but we cannot directly verify its status. For example, Google's servers are an example of beneficial hosts that would stand out in such an analysis unless accounted for.

However, others have the ability to more closely analyze payload, through, for example, the analysis of malware collected in a honeypot. Several organizations make a list of known malicious hosts available to the community. For our purposes, we use the lists published from DShield and SRI Malware Threat Center to create a large set of possible malicious hosts. They provide non-obfuscated IP addresses, which we anonymized [10] similarly to the IP addresses in our flow data (cf. Section 2). More specifically, we use DShield's recommended block list [14], with 20 subnets and the *Most Aggressive Malware Attack Source and Filters* [15] and *Most Prolific BotNet Command and Control Servers and Filters* [16], 30 day lists, from SRI. The latter two contain about 400-500 hosts together.

We leverage these host classifications to create a set of known malicious hosts, $\mathcal{M}^F$. We use the following definition for malicious hosts and flows.

Definition of a malicious host A host, x, visible in our traffic capture, is defined as being *malicious*, if $x \in \mathcal{M}^F$. All such hosts are added to the *malicious set*, $\mathcal{M}$.

Definition of a malicious flow A flow, f, with endpoints f_s and f_d is defined as being *malicious*, if either $f_s \in \mathcal{M}$ or $f_d \in \mathcal{M}$.

We usually downloaded the external malicious host lists in conjunction to the general data collection, and then aggregated them ($\mathcal{M}^F$). Note that we never reclassified these hosts; if they have been deemed to be malicious at one point during the data collection period, they were malicious the whole period. We chose this policy for its simplicity, and given the relatively short time span of the collection period, we do not find this to be a problem. The original data may also lag slightly in time (a host is only discovered as malicious after a series of activities), and by treating it as malicious the whole time we do not miss any of its initial behavior.

The set $\mathcal{M}^F$ contains 25,900 potential hosts, where we on average saw activity from about 5.0% of these hosts in the outgoing traffic and 4.6% of these hosts in the incoming traffic. The sets have about 30% overlap, i.e. of the malicious hosts seen on the outgoing link only 30% of the same sources were also present on the incoming link the same collection date. We would like to emphasize that no hosts inside SUNET belonged to $\mathcal{M}$. The hosts in this set were thus all outside SUNET.

The resulting list of malicious hosts allows us to find *malicious flow patterns* that in turn can be used for a larger analysis on a wider set of hosts. Also, concentrating on the hosts in the malicious group facilitates the analysis as it is easier to find patterns in this smaller subset. Certain attack patterns, such as denial-of-service attacks, cause by their very nature a very large footprint on the flow data and can easily be found (see for example [17]). However, we are also interested in behavior that is not so large scale, and that is part of the reason why we bootstrap our analysis with the *malicious set* $\mathcal{M}$.

Finally, our set of malicious hosts is quite restrictive, e.g. a host needs to display quite aberrant traffic to be on the block list from DShield. There are probably many other hosts that are malicious but are not in our malicious set, as e.g. the IRC C&C server described in Section 3. Thus, we expect that certain patterns found for the malicious hosts will also be applicable to some, what might seem to be, *normal hosts*.

5 Analysis of Malicious Host Behavior

We use the set of *Malicious Hosts*, $\mathcal{M}$, defined in Section 4, to discriminate normal flows from malicious ones. We divide the analysis into two parts. First, we look at overall characteristics of the malicious flows and discuss large malicious footprints. We then describe two particular patterns found by analyzing the traffic flows to malicious hosts on the *outgoing* link.

5.1 Characteristics of Malicious Flows

In Table 5, we show the average fraction of malicious flows, i.e. the number of malicious flows divided by all flows averaged over the data collection period. We note that for incoming traffic, we seem to have more malicious flows over TCP while for outgoing traffic, ICMP flows are dominating. We can explain this fact by previous observations on data from an older generation of SUNET[3] showing that the majority of anomalies (including unsolicited network scanning) originates outside SUNET, i.e. on the main Internet [13]. Table 5 once more confirms these earlier observations with higher numbers of incoming TCP flows, many of them probably SYN probing attempts.

Table 5: Average fraction of malicious flows per protocol.

	Incoming Link	**Outgoing Link**
TCP	0.35%	0.05%
ICMP	0.02%	0.16%
UDP	0.04%	0.01%

Since possible responses to such unsolicited probes are important to understand for the following analysis, we briefly outline them here. Basically, we can differentiate between four scenarios following incoming SYN probings or connection attempts: *i)* replied by SYN/ACK packets,[4] i.e. connection establishment (which should be rather rare for unsolicited scanning events); *ii)* unreplied, e.g. by firewalls; *iii)* replied with a RST response from host sockets;[5] and finally *iv)* replied with *type 3* (*net/host/port unreachable*) ICMP messages from network or end nodes.

Discussion: The larger number of outbound malicious ICMP flows is likely to be an artifact of the unbalance caused by incoming unsolicited TCP probes. In fact, 75% of the outgoing malicious ICMP flows are of *type 3 – destination unreachable*, which is an overrepresentation compared to around 50% *type 3* messages when analyzing all flows.

[3] GigaSUNET, a ring architecture, was in 2007 replaced by OptoSUNET, a star architecture.

[4] SYN/ACK packets are typically larger than 40 Bytes, i.e. 20B IP header, 20B TCP header, up to 20B TCP options header, no payload.

[5] RST packets are normally exactly 40B, i.e. 20B IP header, 20B TCP header, no payload.

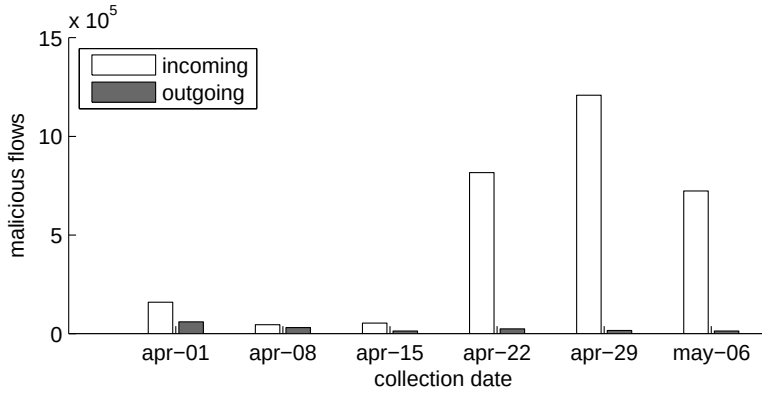

Fig. 3: The figure shows the number of malicious flows detected each data collection day.

Incoming Malicious Traffic: The High-Hitters

In Figure 3, we show the number of *malicious flows* for each data collection day. In general, we observe more incoming than outgoing malicious traffic. We can also see an inflation of incoming malicious flows by a factor of 15–25 on the last three days. This increase stems from a very small number of IPs responsible for the majority of malicious flows. In the following, we define *high-hitters* as source IPs outside SUNET responsible for more than $4k$ malicious incoming flows during one day. Disregarding the high-hitters further discussed below, we found a quite stable amount of incoming malicious traffic during all days, consisting of between $25k$–$35k$ flows stemming from between 1,108 and 1,349 malicious hosts per day.

On April 1, we observe one high-hitter, responsible for 83% of the incoming malicious flows on this day. This host sent UDP packets to $132k$ different hosts inside SUNET on port 1434 with 404-Byte-sized packets during a period of 21 hours. The port number and packet size suggests that this host tried to spread the *Sapphire worm* [18].

In the data from April 8 and 15, we observe one high-hitter that was at both dates responsible for about 33% of incoming malicious flows, a rather moderate traffic density compared to high-hitters found other days. Flows from this host (to 40 hosts inside SUNET) were probably DNS responses, since they came from UDP port 53 with packet sizes of typical DNS answers (around 120 Bytes) and there were corresponding DNS queries (around 70 Bytes) from a few SUNET hosts found on the *outgoing link*. We suspect that this host might have been involved in some sort of *DNS poisoning attack* [19].

On April 22, we observe as many as five high-hitters, responsible for 97% of incoming malicious traffic. Three of these high-hitters (generating 44%, 10% and 10% of the flows, respectively) attempted to connect and login at large IP address ranges of up to $300k$ hosts via either SSH (TCP port 22) or VNC (TCP port 5900) during a couple of hours. The remaining two high-hitters (22% and 11%) also talked to large IP ranges (around $60k$ hosts) without fixed destination port numbers, but rather with fixed TCP source ports of 31414 and 1723, respectively. This would indicate that we actually observe return-traffic from SUNET to these hosts on these port numbers, but we have to further investigate this behavior for its significance.

On April 29, there were three high-hitters, together responsible for 97% of the incoming malicious flows. The main host (59%) was active during the entire 24 hour

period and connected to $107k$ hosts on five different proxy port numbers (e.g. TCP 8080, 3128, 1080, 9415) from port 6000, which is a scanning behavior also observed elsewhere [20]. The other two high-hitters (23% and 15%) showed similar behavior to the two unexplained high-hitters on April 22, with random destination port numbers but fixed source ports of 14700 and again 31414.

On May 6, there was only one high-hitter, responsible for 96% of all incoming malicious flows. This host was scanning on TCP port 1433 (MSSQL), which is known for many vulnerabilities. Interestingly, this scanner also used a single source port number of 6000, and is from the same $\backslash 24$ network as the main high-hitter on April 29.

Discussion: The data basically include a quite constant level of *background radiation*, as also observed elsewhere [21,22]. However, at the same time we observe *transient high-hitters* with varying traffic density. These outstanding, special events complicate determination of *regular traffic patterns* and highlight the importance of longitudinal measurements spanning time, allowing us to differentiate between the transient high-hitter traffic from the constant background radiation in our analysis.

5.2 The Ubiquitous Malicious Hosts

We also decided to investigate how long the malicious hosts were active and the behavior of the most active hosts. We used $\mathcal{M}_{out}$, i.e. the set of all visible malicious host found in the outgoing traffic. We then counted how many collection days these malicious hosts could be found (see Figure 4). As can be seen, a majority of the hosts were only visible a single day during the collection period. A little more than 20% were visible for two days, and only about 3% were visible all collection days. This should be compared with the pie chart to the right in Figure 2.

Discussion: The behavior of the malicious hosts was different from the behavior of all hosts (cf. Figure 2). For example, there were more malicious hosts active all six data collection days, as compared to all hosts. However, we believe this may be an artifact of using a predefined malicious set, i.e., for a host to be blacklisted it must exhibit malicious behavior over a period of time.

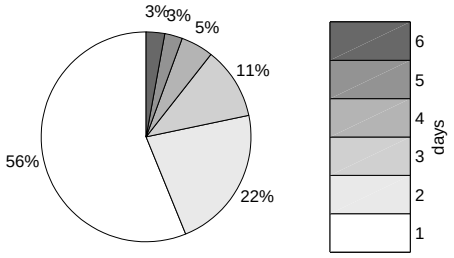

Fig. 4: The number of days the malicious hosts were active in the *outgoing* traffic.

Below we further investigate the traffic flows for the malicious hosts that were present all six data collection days. Even though they only made up 3% of all the malicious hosts we detected, they still stand for 26% of the malicious traffic we found on

the outgoing link. We concentrate on the two most prolific hosts and their traffic pattern over TCP, below referred to as *host-A* and *host-B*. As these flows are outgoing, the flows originated at an internal host (marked as source in the packet). Either the traffic is in *response* to earlier connection attempts by the malicious host or it is *unsolicited traffic* sent directly to the malicious host. We believe the pattern for *host-A* described below is of the former type, while the patterns seen concerning *host-B* is of the latter type.

Massive Connection Attempts: The Scanner

Host-A was the most active ubiquitous host in terms of number of flows over TCP, being part of 3,593 flows over the whole collection period with 2,904 distinct hosts inside SUNET. Many of these hosts inside SUNET were clustered into portions of $\backslash 24$ subnets, and there were usually only 1–2 flows between two distinct end points. Most of the flows (94%) were directed towards destination port 6000 on *host-A*.

For example, from the subnet with the most TCP flows toward *host-A*, we found flows from 123 distinct hosts. Looking in detail at the flows, there was exactly a single packet from each host to *host-A* of the form:

```
srcSubNet.host:2967 host-A:6000 packets:1 Bytes:40
```

Clearly, this is a RST packet in response to a scan. This particular scanning technique, using port 6000, has also been seen elsewhere [20] even though the tool behind it is not completely understood. We also saw this kind of behavior among the high-hitters on the *incoming* link (see Section 5.1).

It is particularly interesting that *host-A* probed different services over the collection period. A majority of the captured flows came from April 1 (69%), where *host-A* probed port 2967 (from 6000). Similarly, on April 8, port 2967 was probed. On April 15, we see the first sign of a new target; 51% of the captured flows were the result of a scan to port 2967 but 39% also targeted port 135, 6% port 1617, and, finally, 4% port 3230. On April 22 the shift was larger still; 21% targeted port 2967 while 79% targeted port 135. On April 29, we can only see a single connection: one RST flow from source 2967 to destination port 6000. On May 5, the malicious host was again more active. At this particular day, only a single connection went from 2967 to destination port 6000, while 99% instead involved port 135. Thus, the potentially vulnerable target port shifted over the collection period, where first port 2967 and in the end only port 135 was probed.

Table 6: Pattern for a possible secondary return and infection.

src	dst	sport	dport	pkts	bytes	date	time
src_1	host-A	2967	6000	1	60	2010-04-22	04:09:16
src_1	host-A	2967	1143	927	48,212	2010-04-22	04:09:21

The second interesting observation of *host-A*'s behavior is the following; the malicious host immediately tried to connect (and infect?) hosts that seemed to have the appropriate service running. As we said above, most of the traffic was actually a single packet with size 40, i.e. a RST packet. What is interesting is when the SUNET host replied with other packet sizes. In Table 6, we list one such example taken from the data captured on April 22. The first flow summarizes the probing attempt by *host-A* but the response we see is *not* the typical RST packet, i.e. the connection seemed to

be accepted.[6] Within 5s, *host-A* returned and opened a connection to the SUNET host and then data were actually exchanged, possibly being malicious code. We see similar behavior on all days; if the first attempt did not elicit a RST packet, there was a follow-up flow in almost all cases. For example, on April 1, there was a new flow, on average within 12s, in 25 of the 27 cases where the SUNET host replied with a packet of size 44.[7] This tells us something about the scanning software. In the first pass, it tries to connect from a standard port and it probably blasts out packets. If the service is not refused, it returns within 10–12s and reconnects through other ports.

Discussion: Summarizing the behavior of *host-A*, we first see that the scanner remains constant over the data collection period despite it being blacklisted. Apparently, the owner did not feel it is worth changing the IP address (because few home users use blacklists?). Second, *host-A* was actively monitored and supervised as we can see from its shifting probing profile over the collection period. Third, the return after a successful probe happened within seconds, either for further data collection or an infection attempt. As future work, it would be interesting to monitor these possibly infected SUNET hosts for their post-infection behavior.

Temporal Patterns: Connecting to the Malicious Server

Host-B was the second most active malicious host that was also present on all connection days. We found 972 flows involving this host coming from 27 distinct sources in the outgoing data. These flows do not seem to be part of a scan; for many of these outgoing flows, a few packets were sent from a non-privileged port from the host inside SUNET to a few very specific ports on the malicious host. We did not at all see similar scanning behavior as with *host-A*. Interestingly enough though, these flows to *host-B* sometimes seemed to follow *temporal patterns*.

We analyzed the traffic patterns based on their time properties from four of the hosts inside SUNET communicating with *host-B*, shown in Figure 5. Each subgraph (with one exception) shows all flows between a single host inside SUNET to a specific destination port on *host-B*. The connection index n in the graph represents flow n (ordered chronologically), which is then plotted at $(t_n - t_{n-1}, n)$. That is, the x axes represent the time differences between two consecutive flows, while the y axes simply index the flows ordered by time. Let us look at each of the subplots separately. We order them from top to bottom, left to right, according to the number in parenthesis in the figure text along the x axes.

In Subplot 1 in Figure 5, we show one of the time patterns we found. Here, the source host connected to *host-B*, port 6969, about once every 43min. Now and then, such a connection was immediately followed by another flow (i.e. through a new source port on the host inside SUNET), probably a reconnect after a failed first attempt. This particular pattern existed over two collection dates, where the flows from the second day is marked with unfilled markers.

Similarly, in Subplot 2, we found a consistent pattern but with a period of 30min instead of 43min as in Subplot 1. In this graph, we have also added dotted lines between two consecutive connections to make the connection pattern more visible. The SUNET

[6] Note that SYN/ACK responses are, due to TCP option headers, typically larger than 40 Bytes.
[7] The missing two cases may be an effect of the routing bias explained in Section 2.3.

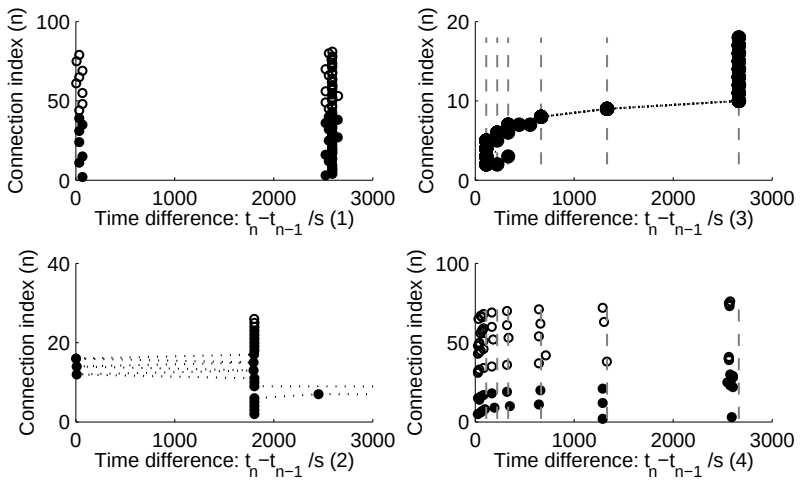

Fig. 5: The figure shows temporal communication patterns to the malicious *host-B*.

host connected to *host-B*, port 8000, about every 30min with a few exceptions. Three times, the "ordinary" connection was immediately followed by a new connection. Two times[8] the reconnections took more than 30min to establish. Again, the pattern lasted over two collection dates spaced a week apart, and the flows from the second day are marked with unfilled markers.

Discussion: The patterns seen for the hosts in Subplot 1 and 2 imply relatively simple programming, i.e. a regular refresh or a keep-alive signal. The hosts inside SUNET may contain malware, meaning that these patterns would indicate an attempt to "call home" by the malware to the blacklisted *host-B*. However, sometimes other services are also running on the malicious hosts and the seen pattern could be the result of a more regular service. Especially interesting is the pattern length; interpolating, it would seem that the keep-alive signal was present over a week.

In Subplot 3 we can see a complex back-off pattern with $t_1 = 111s$, $t_2 = 222s$, $t_3 = 333s$, $t_4 = 666s$, $t_5 = 1,332s$, $t_6 = 2,664s$, i.e. $t_n = \sum_{i=1}^{n-1} t_i$ for $n > 2$. The vertical dotted lines are added at these *anchor times* for ease of reading the figure. The source tried to connect to *host-B* with a total of about 16–17 distinct flows, with about 3–4 connections spaced 111s apart. The following connection then came after $222s$ followed by one at $333s$, $666s$, $1,332s$, and finally about nine flows spaced $2,664s$ apart. What makes this particular pattern stand out, apart from its complexity, is that the source connected to *host-B* on several distinct ports (6969, 8000, 8080), always following the exact same pattern, seldom being off even a full second from the *anchor times* described above. Moreover, this very same host also tried to connect to *six* other hosts, using the exact same connection pattern but with different destination ports (80,

[8] In Subplot 2 we have one outlier at about 38,293s that is not shown (the dotted lines hint to its existence). In Subplot 4, we have five outliers not visible in the graph.

2710, 6997, 6969, 9999, 60500). Thus, in total the host exhibited the very same pattern in *twelve* distinct cases and they have all been superimposed in Subplot 3. Even though Subplot 3 in reality contains 12 graphs, we can see that the pattern in each of these graphs is so similar to the others that each individual point is plotted on top of another and it is easy to distinguish the overall structure. There are a few errant points in the beginning but towards the end the sequence is stabilized. Most of the flows that make up this pattern, contained three packets with a total of 152 Bytes. These 12 patterns appeared within four minutes of each other, which might indicate a common event triggering their initialization. They may also have contained more points than shown in the graph, but any subsequent point is beyond our 24h data collection period.

Finally, in Subplot 4 we show a similar pattern to that found in Subplot 3. We could only find the single host in Subplot 3 with a back-off pattern within a second of the *anchor times*. However, we found a few other instances where the pattern is somewhat similar. One such example is shown in Subplot 4, where we again show flows from a subsequent collection day with unfilled markers. In contrast to Subplot 3 (where 12 similar patterns are superimposed), Subplot 4 contains a single pattern, i.e. one host that used different source ports to connect to the same destination port (8080) on *host-B*. In contrast to the patterns in Subplot 3, we can see that this host was repeating the pattern over and over again, lasting over a week, i.e. two data collection dates.

Discussion: The time properties for these two hosts represent a more complex programming logic, implying that the programmer chose this particular algorithm for a reason. As discussed above, these patterns could indicate an attempt to "call home" by the malware but it could also be part of a more regular service.[9] The patterns were probably triggered by some event, as all 12 occur within four minutes of each other. Given the exact nature of the pattern displayed by the host in Subplot 3, we have described it at several mailing lists but without conclusive responses as most people suggested a full packet capture for further analysis – which is not possible for us due to privacy concerns. We have also been unable to find other hosts within our data that follow such an exact pattern as displayed by this host. This means that regardless of the pattern being the result of a regular program or malware, it is not widespread.

6 Related Work

As already outlined, malicious traffic can be studied by several orthogonal methods, such as *distributed sensors*, *honeypot networks*, *network telescopes/darknets*, and *large-scale passive measurements*. The relation of the first two methods to our work has already been discussed in Section 1. They introduce a serious bias, as the users obviously care about security, and they are not very suitable for analysis of *real* user responses.

Given that *network telescopes* monitor large, unused IP address spaces, they see traffic that by its very nature should not exist [23]. Even though network telescopes have been used for extensive studies of worm outbreaks [24] and for general characterization of background radiation [25,26], they are only traffic sinks and do not respond genuinely to incoming traffic.

[9] For example, one of the destination ports found (6969) may be associated with bittorrent.

Our approach, passive measurements on large-scale links, is generally viewed as the best way to study Internet traffic, as it includes real behavioral responses from a diverse user population. Others made use of observations of connection properties to study general characteristics of scanning traffic both on campus links [21] and backbone links [27]. Also one of the authors of this paper previously quantified unsolicited traffic by simple heuristic methods utilizing connection patterns [17,22]. Malicious traffic (i.e. scanning and DDoS attacks) was observed to be the main reason for short, unidirectional one-way flows on both campus and backbone links [28]. Rehák et al. [29] uses NetFlow data to fine-tune an Intrusion Detection System (IDS) by periodical insertion of challenges (or fault injections).

In contrast, in this paper we have observed and analyzed traffic patterns of malicious hosts, describing their changing behavior over the data collection period. Behavioral analysis of malware is also possible by reverse engineering [30], and we consider such approaches complementary to ours; reverse engineering requires a significant effort but may yield an exact analysis of the malware in question but with our measurements a wide range of behavioral patterns can directly be observed. Furthermore, reverse engineering can never directly answer certain questions, such as how widespread a certain malware is, but this is a property we can measure.

7 Discussions and Conclusions

We have shown that we can use anonymized flow data to discover and analyze malicious flow patterns, a result useful for network researchers and practitioners interested in security related topics such as intrusion detection. Some attacks leave such a big footprint that they are visible even in summaries of large traffic aggregates, as the C&C server described in Section 3. To detect more obscure patterns, though, we bootstrapped our analysis with a predefined set of malicious hosts, and analyzed their behavior from a large-scale perspective based on Internet backbone data. Each finding is *discussed* in detail within the paper. For example, we showed the need for longer-time measurements to be able to separate the *transient high-hitters* from the background traffic. Despite being blacklisted, we found some malicious hosts that stayed active over the whole measurement period. One of these hosts seems to automatically scan and possibly infect vulnerable hosts within seconds, but is most likely under active human supervision as its scanning profile shifted over time. In contrast to many previous measurement methods, we went beyond the analysis of simpler attribute aggregates (such as top source port, etc.) to also include a *temporal analysis of communication patterns*, originating from hosts within SUNET to a malicious host. We found both simple refresh logic and complex back-off patterns, sometimes lasting over a week. The latter patterns are not easy to discover because they do not leave large footprints in traditional traffic summaries.

Summarizing, there are disadvantages with using anonymized flow data in that it is rather difficult to both interpret the data and to validate the result. However, such data would otherwise not be available for analysis and they can still offer many valuable insights, not possible with complementary methods. The findings in this paper are just the first look at the data, which we are still expanding by regular weekly measurements (ongoing since April 2010). By improving the selection of the malicious hosts, both by using collected information from locally installed honeypots with access to full payload

and a more automatic classification of hosts and malicious traffic [31], we expect more detailed and conclusive results in the future.

References

1. L. Corrons, "Computer Threat Trend Forecast for 2010," http://pandalabs.pandasecurity.com/computer-threat-trend-forecast-for-2010/, Dec. 2009.
2. R. S. Mueller III, "Major Executive Speeches, RSA Cyber Security Conference," http://www.fbi.gov/pressrel/speeches/mueller030410.htm, 2010.
3. Symantec, "AntiVirus, Anti-Spyware, Enpoint Security," 2010, http://www.symantec.com.
4. McAfee, "Antivirus, IPS, Firewall, Web Security," 2010, http://www.mcafee.com.
5. The Honeynet Project, "Honeynet Project Blog," 2010, http://http://www.honeynet.org.
6. NoAH, "European Network of Affined Honeypots," 2010, http://www.fp6-noah.org.
7. DShield, "Cooperative Network Security Community - Internet Security," 2010, http://www.dshield.com.
8. SANS, "Internet Storm Center," 2010, http://isc.sans.edu.
9. W. John, "Characterization and Classification of Internet Backbone Traffic," Chalmers University of Technology, Doctoral Thesis ISBN 978-91-7385-363-7, 2010.
10. J. Fan, J. Xu, M. Ammar, and S. Moon, "Prefix-Preserving IP Address Anonymization: Measurement-Based Security Evaluation and a New Cryptography-Based Scheme," *Computer Networks*, vol. 46, no. 2, 2004.
11. D. Moore, K. Keys, R. Koga, E. Lagache, and k. claffy, "The CoralReef Software Suite as a Tool for System and Network Administrators," in *USENIX LISA*, 2001.
12. OptoSUNET, "Core Map," http://stats.sunet.se/stat-q/load-map/optosunet-core,,traffic,peak.
13. W. John and S. Tafvelin, "Differences between in- and outbound Internet Backbone Traffic," in *TERENA Networking Conference (TNC)*, 2007.
14. DShield, "Recommended block list," 2010, http://www.dshield.org/block.txt.
15. SRI International Malware Threat Center, "Most aggressive malware attack source and filters," 2010, http://mtc.sri.com/live_data/attackers/.
16. SRI International Malware Threat Center, "Most prolific botnet command and control servers and filters," 2010, http://mtc.sri.com/live_data/cc_servers/.
17. W. John and S. Tafvelin, "Heuristics to Classify Internet Backbone Traffic based on Connection Patterns," in *Int. Conference on Information Networking (ICOIN)*, 2008.
18. D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "The Spread of the Sapphire/Slammer Worm," CAIDA, Tech.Rep., 2003.
19. S. Friedl, "An Illustrated Guide to the Kaminsky DNS Vulnerability," 2008, http://www.unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html.
20. G. N. White, "What's up with all the port scanning using TCP/6000 as a source port?" 2010, http://isc.sans.edu/diary.html?storyid=7924.
21. M. Allman, V. Paxson, and J. Terrell, "A Brief History of Scanning," in *Internet Measurement Conference (IMC)*, 2007.
22. W. John, S. Tafvelin, and T. Olovsson, "Trends and Differences in Connection-behavior within Classes of Internet Backbone Traffic," in *Passive/Active Measurement (PAM)*, 2008.
23. D. Moore, C. Shannon, G. Voelker, and S. Savage, "Network Telescopes," CAIDA, Tech.Rep., 2004.
24. CAIDA, "Research:Security," 2010, http://www.caida.org/research/security/#PreviousMalware.
25. R. Pang, V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson, "Characteristics of Internet Background Radiation," in *Internet Measurement Conference (IMC)*, 2004.
26. M. Bailey, E. Cooke, F. Jahanian, J. Nazario, D. Watson *et al.*, "The Internet Motion Sensor: A Distributed Blackhole Monitoring System," in *SNDSS*, 2005.

27. A. Sridharan, T. Ye, and S. Bhattacharyya, "Connectionless Port Scan Detection on the Backbone," in *IPCCC*, 2006.
28. D. Lee and N. Brownlee, "Passive Measurement of One-way and Two-way Flow Lifetimes," *ACM SIGCOMM Comp. Comm. Rev.*, vol. 37, no. 3, 2007.
29. M. Rehák, E. Staab, V. Fusenig, M. Pěchouček, M. Grill, J. Stiborek, K. Bartoš, and T. Engel, "Runtime Monitoring and Dynamic Reconfiguration for Intrusion Detection Systems," in *Recent Advances in Intrusion Detection (RAID'09)*, 2009.
30. P. Porras, H. Saidi, and V. Yegneswaran, "An Analysis of Conficker's Logic and Rendezvous Points," Computer Science Laboratory, SRI International, Tech.Rep., 2009.
31. M. Almgren and E. Jonsson, "Using Active Learning in Intrusion Detection," in *17th IEEE Computer Security Foundations Workshop (CSFW 2004)*, 2004.

Webpage URLs in *References* have been accessed July, 2010.