Clairvoyance: Inferring Blocklist Use on the Internet

Vector Guo Li[1], Gautam Akiwate[1], Kirill Levchenko[2], Geoffrey M. Voelker[1], and Stefan Savage[1]

[1] University of California, San Diego
[2] University of Illinois Urbana-Champaign

Abstract. One of the staples of network defense is blocking traffic to and from a list of "known bad" sites on the Internet. However, few organizations are in a position to produce such a list themselves, so pragmatically this approach depends on the existence of third-party "threat intelligence" providers who specialize in distributing feeds of unwelcome IP addresses. However, the choice to use such a strategy, let alone which data feeds are trusted for this purpose, is rarely made public and thus little is understood about the deployment of these techniques in the wild. To explore this issue, we have designed and implemented a technique to infer proactive traffic blocking on a remote host and, through a series of measurements, to associate that blocking with the use of *particular* IP blocklists. In a pilot study of 220K US hosts, we find as many as one fourth of the hosts appear to blocklist based on some source of threat intelligence data, and about 2% use one of the 9 particular third-party blocklists that we evaluated.

1 Introduction

Over the last decade, the use of threat information sharing — commonly labeled "threat intelligence" — has become a staple in any discussion of network defense. Based on the premise that by broadly sharing information about known threats, organizations can better protect themselves, a burgeoning industry has emerged to collect, aggregate and distribute such information [6, 40], largely consisting of lists of IP addresses, domain names or URLs thought to be associated with particular classes of threats (a.k.a., *indicators of compromise*).

However, despite all the promises, it is far from clear how people actually adopt threat intelligence data, especially for proactive traffic blocking, commonly called "blocklisting". Proactively blocking traffic based on threat intelligence data is uniquely attractive to a defender, since, if effective, it can foreclose threats without requiring attention from a human analyst. However, it is also a strong action, and recent work by Li et al. [23] has shown that threat intelligence feeds can be far from comprehensive and may include significant numbers of false positives that might cause an organization to inadvertently block benign sites. Given this, it is important to understand the extent to which network administrators are using such data to block network traffic in practice.

Motivated by this issue, our work seeks to infer if online hosts use threat intelligence IP feeds (IP blocklists) to proactively block network traffic. The principal challenge in pursuing this question is that such decisions are largely invisible: a network choosing to block IP address A or not is indistinguishable from a third vantage point, as this

vantage point does not have access to either the network or IP address A. Moreover, for operational security reasons, few organizations are willing to publicly document the details of their network defenses.

In this paper, we describe an inference technique, based on the IP ID increment side-channel (inspired by previous work focused on censorship detection [11, 29]), to detect network-layer blocklisting. Our design is both specialized to the unique characteristics of IP blocklists (e.g., dynamic, overlapping membership) and is designed to be conservative with respect to common sources of network measurement error (hence a finding of blocking is robust). To evaluate this technique, we test against known ground truth data and then conduct a large-scale pilot study with over 220K U.S. hosts and against 9 popular IPv4 blocklists. In the two cases where network operators were willing to share their blocking configuration with us, they were in perfect agreement with our findings.

Across our pilot study, we identified 4,253 hosts (roughly 2% of the hosts we surveyed), consistently using at least one of the 9 lists that we tested against. We also established that a larger fraction (roughly a fourth) of the hosts we surveyed make use of some form of security-related blocking and reliably block traffic to at least some subset of the IP addresses in our lists. This significant level of security-related blocking is particularly surprising as our pilot study is biased towards older machines with minimal traffic (a cohort that we would not have associated with organizations having an aggressive network security posture).

2 Background

There is a large body of literature concerning the use of various kinds of "threat intelligence" (not always using that term). One popular focus among these is evaluating their effectiveness, including works that analyze coverage and accuracy of spam blocklists [37, 30], phishing blocklists [35], and malware domain blocklists [20]. Others have explored techniques to better populate such lists, including Ramachandran et al.'s work on inferring botnet IP addresses from DNSBL lookups [33], and the work of Hao et al. for predicting future domain abuse [13, 14] (among others). More recently, Thomas et al. explored the value of sharing threat intelligence data across functional areas (e.g., mail spam, account abuse, search abuse) and found limited overlap and significant numbers of false positives [39]. Many of these results are echoed by Li et al. [23].

However, there is comparatively little work focused on understanding how threat intelligence data is being used in practice. Indeed, the literature that exists is primary driven by surveys [31, 34] and not validated by any empirical measurement.[3]

There also has been significant empirical exploration of Internet connection blocking in the setting of Internet freedom and access. Indeed, there are a range of studies that measure connection block in the context of Internet censorship [2, 4, 10, 28, 43], geo-blocking [1, 27, 25], and Tor blocking [18, 36]. Most of these studies rely on vantage points sited in the target networks being studied, and so are not directly helpful in our work. However, recent work by Ensafi et al. [11] and Pearce et al. [29] has removed this requirement using an indirect side channel technique to test connectivity between

[3] One exception is the recent work of Bouwman et al. [7] which has explored aspects of this question through the interview of over a dozen security professionals.

pairs of remote hosts. While our approach differs in a number of ways from theirs, it is inspired by the same idea of using IP ID to infer if a remote host sent an IP packet.

The *IP ID traffic side channel* has been well-known since mid 1990s. In particular, the Identification (ID) field of an IPv4 packet is a 16-bit value in the IP packet header, designed to support fragmentation by providing a unique value that can be used to group packet fragments belonging to the same IP datagram [32]. The simplistic approach using a per-host global counter to ensure unique IP ID values implicitly encodes the *number* of packets sent. Thus, by probing a host multiple times one can use the value of the returned IP ID to infer how many packets have been sent by the remote host *between* the two probes. This side channel has been employed for a wide variety of measurement purposes, including anonymous port scanning [3], host alias detection [38] and enumerating hosts behind NATs [5] among others. While most operating systems no longer use such a simple approach, it is still reasonably common across the Internet. For example, all versions of Windows up to version 7 used the global increment algorithm [19].

3 Methodology

In this section, we first describe our inference technique, using the IP ID side channel (Section 2), that determines if a particular host uses a known blocklist. The intuition here is that if a *reflector* — a host suitable for our technique — blocks all blocklist IPs from one particular blocklist, then it is likely that the particular blocklist is being used for blocking traffic at the network-layer. Next, we detail criteria of suitable reflectors (Section 3.2), and our criteria when sampling blocklist IPs (Section 3.3). Finally, we discuss additional validation measures (Section 3.4) and ethical concerns (Section 3.5).

3.1 Technique Overview

To measure if a reflector is blocking a particular IP from a blocklist, we send a train of packets (here we use SYN-ACK packets) from our measurement machine to the reflector. The packet train consists of packets whose source address is the blocklist IP (spoofed), bracketed by packets whose source address is our measurement machine, as illustrated in Figure 1. If a firewall in the reflector's network blocks packets from the blocklist IP, the reflector will not receive packets with the blocklisted source address. It will only receive packets with our measurement machine's source address. On the other hand, if there is no blocking, the reflector will receive the entire packet train.

In an ideal world, where there is no packet loss during transmission and no extra traffic on the reflector, we expect the reflector to send a RST response for each SYN-ACK packet we send, and we will receive the responses for the SYN-ACK with our measurement machine's source address. The IP IDs of these received RST packets will reflect the number of packets sent by the reflector. If the reflector did not receive the SYN-ACK packets with the blocklist IP as source addresses (being blocked by a firewall), the IP ID sequence in the RST responses will be an increasing sequence without gaps (the "Blocking" case in Figure 1). On the other hand, if the reflector did receive the SYN-ACK packets with the blocklist IP, it would have sent a RST in response to each such packet, incrementing the IP ID counter each time. While we will not see the RST

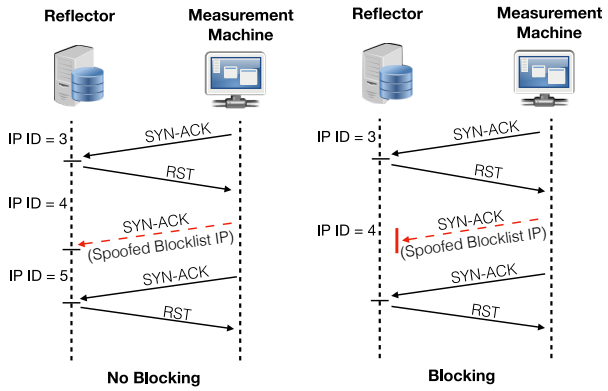

Fig. 1: The basic method to detect network-layer blocking using the IP ID side channel. When there is no blocking in place (left), the measurement machine will see an IP ID gap in two RST responses: the second IP ID will increase by two. Whereas if there is network blocking (right), then the two IP IDs will be consecutive without a gap.

packets sent to the blocklist IP, we *will* observe the increments in the IP ID sequence. More specifically, we would see a gap in the IP ID sequence of packets received by our measurement machine (the "No Blocking" case in Figure 1). These two cases allow us to determine whether a particular blocklist IP is blocked by some network device, such as a firewall, somewhere between the measurement host and the reflector.

In previous works [11, 29], the technique relies on sending spoofed SYN packets to the sites (equivalent to our *blocklist IPs* here), with the source IPs equal to reflector IPs. The sites then reply with SYN-ACK packets to the reflectors. By monitoring the reflectors' IP ID changes during this process, the authors can determine whether the reflectors are blocking the tested sites. To use this strategy, however, one requires both reflectors and sites to be active hosts that reply to SYN/SYN-ACK probes. Unfortunately, in our case there is no guarantee that blocklist IPs will reply to TCP probes. In fact, we found that on average only about 24% of IPs on a blocklist reply to TCP probes. Using only blocklist IPs that reply would dramatically reduce the candidate IPs we can sample from a blocklist, especially for small blocklists that only have a few hundred IPs. We already have many constraints when sampling IPs from a blocklist (Section 3.3), and this extra requirement could leave us with not enough candidates for a measurement.

Therefore, in our technique, we directly send SYN-ACK packets to reflectors, with no involvement of hosts behind blocklist IPs. The disadvantage here is that we cannot detect outbound blocking — wherein the spoofed packet reaches the reflectors but the responses are blocked when going out of the network. Based on our experience talking with several security companies, most customers deploy inbound or bi-directional traffic blocking, so we believe missing outbound blocking is not a major concern.

In this section, we explain how the technique works on a theoretical level. The actual implementation needs to handle potential packet loss and other extraneous traffic at reflectors. We list the full implementation of the technique and false positive and false negative analyses in Appendix A.

3.2 Criteria For Reflectors

At a high level, our technique relies on the presence of the IP ID side channel. Keeping that in mind, listed below are the criteria for suitable reflectors.

- **RST packet generation:** The reflectors must reply with a RST packet to a TCP SYN-ACK packet without an established connection. Hosts that drop incoming SYN-ACK packets without a corresponding SYN packet are not suitable for our methodology. We use SYN-ACK packets instead of SYN because it does not create an intermediate state on the reflectors and the connection is terminated in one go.
- **Shared monotonic increasing IP ID counter:** The reflector should have a monotonically increasing globally shared IP ID counter, so all network traffic from the host uses the same IP ID counter counter and the number of packets generated by the host between two measurements is implicit in the difference of IP IDs.
- **Low traffic:** Our technique relies on a clear observable difference in IP ID. As such, hosts must have low traffic volumes since a high traffic volume makes it infeasible to observe the IP ID changes triggered by our probing packets.
- **No ingress filtering:** We send spoofed packets to reflectors to infer traffic blocking. However, some network providers use ingress filtering techniques and drop packets once they detect the packets are not from the networks they claimed to originate. This filtering would cause our spoofed packets being dropped and give us a false signal of traffic blocking.
- **No stateful firewall blocking:** Some networks deploy a stateful firewall that blocks access from a source IP after receiving too many repetitive packets. One example is to defend against SYN floods [21]. While we try to keep the number of our probing packets as low as possible, if our spoofed packets trigger such firewall rules and then we are blocked by the firewall, we will incorrectly conclude that the reflector uses a blocklist to block that IP.

Our goal is to discover if online hosts are using IP blocklists to block traffic. But when looking at the problem on a global scale, there are many policy related reasons why a host blocks network traffic, such as censorship. These alternate sources of blocking could disrupt our experiments. To simplify the problem, and for ethical considerations, in this paper we only test the hosts located in the United States.

3.3 Sampling Blocklist IPs

To determine if a reflector uses a particular IP blocklist, we use a sample of IPs from a blocklist, as it would be infeasible for us to test all blocklist IPs. Further, to obtain a definitive signal from our experiment, we need to adhere to the following constraints when sampling blocklist IPs to avoid possible noise:

- **Exclusive:** A blocklist can share part of its contents with other blocklists. To reasonably infer whether a reflector is using a specific blocklist, we need to test with IPs unique to that blocklist — IPs that are only in this blocklist but no others.
- **Stable:** IPs on a blocklist change over time. To reliably measure if a reflector blocks IPs from a certain blocklist, we need the sampled IPs to stay in the list throughout one experiment. This cannot be enforced beforehand, so we discard the cases where a blocklist IP does not remain on the list for the duration of the experiment.

- **Routable:** IP blocklists can contain unroutable IPs [23]. Sending packets with an unroutable source address results in a large portion of packets being dropped, as we have observed (which could potentially happen at end ISPs or transient links). Packet drops due to unroutable IPs would create noise in the experiment. Therefore, when sampling IPs from blocklists we ensure that the IPs are routable.
- **Geo-location diversified:** Besides blocklisting, another common reason for traffic blocking is geo-blocking, where a host blocks all traffic coming from a certain country or region. To minimize the effect of geo-blocking, we prioritize IPs that are from the United States when sampling IPs, assuming a host in the US will not geo-block traffic from the US. For IPs in other countries, we try to increase the diversity of IP locations, making sure the sampled IPs are not concentrated in only a few countries when possible.
- **Not from reflectors' network (AS disjoint):** We observed that not many networks have implemented ingress filtering (we saw less than 2% of the total hosts we scanned showing this behavior). However, many networks drop spoofed packets when the spoofed source addresses are within their own network. So when selecting blocklist IPs, we make sure that these IPs are not from the same ASes as one of our reflectors.

3.4 Control Group

To further validate our technique, every time we test a set of blocklist IPs against each reflector, we also include a control group of 20 randomly chosen IPs that are BGP routable, geo-located in the United States and not blocklisted (see Section 4.2). The control group represents a random set of IPs that are unlikely to be blocked in bulk by a reflector. We use US IPs to avoid the potential problem of geo-blocking. If a reflector does block a significant fraction of control IPs, it is probably because the reflector is not suitable for this technique (one reason can be that our ingress-filtering step did not catch these IPs), and we should discard all the results associated with this reflector.

3.5 Ethical Considerations

In our experiments, we send spoofed packets to reflectors impersonating traffic from other IPs to infer the presence of network-layer blocking based on IP blocklists. A key ethical concern with this kind of measurement is the extent to which either receiving such packets or being seen to have received such packets would put the recipients at undue risk. Indeed, this is particularly problematic in censorship measurements [11, 29] because of the potential to inadvertently cause a host to be associated with content that is politically dangerous in their country. However, our work operates in a context that is substantially less risky, and we have further designed multiple aspects of our protocol to minimize the likelihood of risk. In particular, our methodology incorporates the following approaches to minimize risk:

Restriction in Scope: We have specifically restricted our measurements to only reflectors within the United States, which affords relatively robust free speech rights and considerable transparency around criminal proceedings. Indeed, from our conversations

Category	Count
IP Addresses	222,782
/24 Count	128,712
Autonomous Systems	3,371

Table 1: The number of reflectors (IP addresses) identified in the United States, and the corresponding count of /24 prefixes and Autonomous Systems.

with both network operators and law enforcement, we are unaware of a realistic scenario where the mere receipt of a packet has led to criminal or civil liability.

Conventional sources: Unlike in censorship studies, the source IP addresses being spoofed in our measurement are those that have been used to mount wide-spread abusive activity such as spamming, port scanning, etc. and these represent precisely the kinds of traffic that a typical host on the Internet would *expect* to receive.

Inbound, connection-free probes: Our measurements are constructed to be inbound only and connection free; that is, a network monitor could witness traffic consistent with an *external* scan of one of their hosts, but will never witness a completed connection or any data transmission. From our discussions with network operators and network security vendors, we could not identify a scenario where the mere receipt of the packets we send would be sufficient to drive an incident response team to action.

Minimal use of end-host resources: Our scans are purposely constructed with SYN-ACK packets to ensure that no state is created on the reflector. Moreover, our peak probing rate per reflector is 6 min-sized packets per second (see appendix for more details). But even that rate only persists for two seconds in each test, and in the following pilot study, we probe each reflector no more than once every 3 minutes.

4 Pilot Study Implementation

With the technique discussed in the previous section, one can then infer if an online host (reflector) satisfying the selection criteria outlined above is blocking traffic using a specific IP blocklist. To evaluate our inference technique, we conducted a pilot study over a large number of reflectors to infer their blocklist usage. In this section, we explain in detail the implementation of our experiment, including reflector selection, blocklist selection, sampling IPs from blocklists and measurement setup.

4.1 Reflector Selection

We start our selection of reflectors using a snapshot of Censys [9] scanning data from November 8, 2019, consisting of over 40 million IPv4 hosts with open ports in the US. We then send multiple probes to each host targeting an open port from different source addresses, checking the IP IDs of responses to identify the ones with the IP ID side channel. We further run tests to make sure they meet the criteria listed in Section 3.2 (see Appendix A). If one host has multiple open ports, we randomly pick one to probe.

We identified 222,782 IP addresses in the US that meet our criteria. For the purpose of this paper, we treat each individual IP address as a distinct reflector. Table 1 counts these addresses at different network aggregations. By construction, the set of reflectors we use will necessarily have certain biases. To understand what fraction of networks of potential interest to others this might cover, we queried the Alexa top 100K domains as of Dec. 17th, 2019 for their A records and MX records and obtained their corresponding IP addresses. Of these, we identified a total of 94,846 IPs that are located in the US, covering 34,083 /24s. While we made no attempt to find reflectors in these networks *a priori*, our selection methodology identified at least one reflector in 16.9% of these /24s. When only looking at the top 10K domains, our data set covers 13.2% of US /24s.

We also checked the WHOIS record of each reflector and identified all hosts associated with education institutions. In total, our data set includes 4,370 education IPs, ranging across 181 different institutions, and covers 40 out of the top 100 US universities based on the US News ranking [42]. Thus, while there may be networks without a suitable reflector for one reason or another, our technique is applicable to a large number of existing networks.

4.2 Choosing Blocklists and Sampling IPs

For the pilot study, we choose candidate blocklists from public IPv4 blocklists. We use the FireHOL IP blocklist collection [12], which aggregates over 100 public IP blocklists every day. However, we cannot reasonably test against all the blocklists and so, for the purposes of this paper, we select the most popular public IP blocklists and then do a more detailed measurement on them.

For each of the public IP blocklists, we sample five IPs (using the criteria in Section 3.3) from each list and test how many reflectors block all sampled blocklist IPs in each blocklist. The goal of this step is to roughly estimate how widely used these blocklists might be, so that we can pick the most prevalent ones for more detailed measurements later in Section 5. We repeat the measurement twice and select the top 9 blocklists:

1. **Spamhaus DROP**: Spamhaus Don't Route Or Peer Lists
2. **Spamhaus EDROP**: An extension of the Spamhaus DROP list
3. **DShield Top Blocklist**: DShield.org recommended top 20 /24s to block
4. **ET Compromised**: EmergingThreats.net recorded compromised hosts
5. **Snort IP Filter List**: labs.snort.org supplied IP blocklist
6. **BDS IP Ban List**: Binary Defense System ban list
7. **Feodo IP Blocklist**: Abuse.ch Feodo tracking list
8. **Blocklist De Blocklist**: Blocklist.de blocklist IPs
9. **Tor IP Blocklist**: IPs that belong to the Tor network (not just exit nodes)

When sampling IPs from blocklists to test, we use the criteria listed in Section 3.3. To find the exclusive IPs on each blocklist, we use the public IP blocklists collected by FireHOL, as mentioned earlier, and calculate the unique part of each target blocklist. For the stable IP requirement, we collect all the target blocklists hourly, and ensure the sampled IPs are in the blocklist through the duration of the experiment. To satisfy the routable requirement, we use daily RouteView data [41] to identify BGP routable IPs.

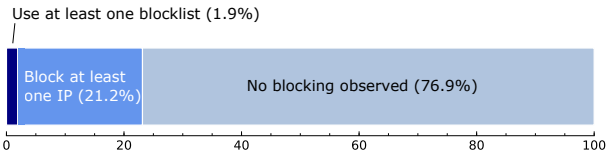

Fig. 2: Breakdown of reflector blocking based on three experimental runs. We identified 4,253 reflectors that use at least one blocklist (Section 5.1). We also found a large number of reflectors blocking at least some IPs in blocklists (Section 5.2).

For geo-location diversity, we use NetAcuity [26] to make sure for each experiment the sampled IPs cover as many different countries as the data allows.

4.3 Measurement Setup

Having selected the reflectors and blocklists, we can now conduct the experiment to infer which reflectors use which specific blocklist.

For a particular experimental run, we randomly selected **25** IPs from each blocklist that satisfy the requirements defined in Section 3: exclusive, stable, routable, geo-diversified, and AS disjoint. Then we evaluated the blocking behavior for all 220K reflectors against the 225 blocklist IPs sampled from the 9 blocklists. To handle cases where reflectors might take time to update and start blocking the newest IPs on the blocklist, we ensure the sampled IPs have appeared in the blocklist at least 2 weeks before our experiment. During post-processing, we remove blocklist IPs from consideration that did not remain on the list for the duration of the experiment. Furthermore, we conducted three experimental runs, each time using a different set of 25 IPs from each blocklist. We then conclude that a reflector is using a blocklist if and only if all experiment runs show that it blocked all the sampled IPs from that blocklist.

We conducted our measurements from December 3–23rd, 2019. During this period, we tested in total 96,067,051 distinct `(reflector, blocklist IP)` pairs. (In the first two experiments, we tested against all reflectors. In the last experiment, we only tested the ones that have shown blocking behavior in the first two tests.) Among these pairs, 894,570 pairs display a clear signal indicating "blocking".

5 Pilot Study Overall Results

Figure 2 presents the overall blocking behavior of all 222,782 reflectors we tested partitioned into four categories: those reflectors that we conclude use at least one of the public blocklists (1.9%), reflectors that block at least one blocklist IP (21.2%), and reflectors that do not block any blocklist IPs (76.9%). Note that given the attributes of hosts to be reflectors, such as running old OS versions, it is not surprising a large percentage shows no blocking of the blocklist IPs: they already have attributes anti-correlated with high degrees of security hygiene. The following sections explores each of these categories of reflector blocking behavior in more detail.

Blocklist (abbr.)	**Reflectors**	**/24s**	**ASes**
Spamhaus DROP (DROP)	4,142	1,782	50
Spamhaus EDROP (eDROP)	1,272	362	25
DShield Top Blocklist (DTop)	223	69	18
ET Compromised (ET)	116	58	15
BDS IP Ban List (BDS)	85	41	3
Feodo IP Blocklist (Feodo)	64	26	16
Snort IP Filter List (Snort)	52	20	11
Blocklist De Blocklist (DE)	36	18	8
Tor IP Blocklist (Tor)	24	9	8
Total Unique	4,253	1,827	77

Table 2: Breakdown of reflectors we conclude using each of the nine blocklists.

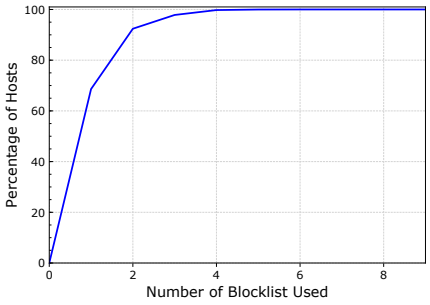

Fig. 3: CDF of the number of blocklists used by reflectors.

5.1 Reflectors Using Blocklists

We identified 4,253 (1.9%) reflectors that use at least one of the 9 public blocklists. Table 2 shows the number of reflectors using each of the nine different blocklists, as well as the number of unique /24s and ASes those reflectors appear in. Spamhaus DROP is by far the most popular blocklist in our collection, followed by Spamhaus EDROP. The remaining blocklists have a comparatively small number of reflectors using them. Since many aspects of our method and experiment make conservative choices, these results should be considered a lower bound.

Figure 3 shows the cumulative distribution of the number of blocklists these reflectors use. For the 9 public blocklists we studied, over 68.6% use just one blocklist, 23.8% use two or more, and 7.6% use three or more. One reflector used 6 of the 9 blocklists.

For these reflectors, though, there are interesting patterns to the multiple blocklists used. Figure 4 shows the use of multiple blocklists with a heatmap. Rows and columns correspond to blocklists, and each cell of the heatmap shows the fraction of the reflectors using the blocklist in row R that are also using the blocklist in column C. For example, the first cell for ET Compromised shows that 78% of the reflectors that use ET also use the Spamhaus DROP blocklist. Diagonal cells are 1.00 since they show blocklists compared with themselves. Blocklists are ordered in the same order as in Table 2.

The first cell of the Spamhaus EDROP row indicates that all reflectors that use Spamhaus EDROP also use Spamhaus DROP. Since the eDROP list is an extension of the DROP list, the behavior is strongly consistent with expectations. Moreover, the many significant values in the first two columns show that reflectors that use any of the other blocklists very often also use Spamhaus DROP and eDROP. At least for the hosts that we select for, these results underscore the popularity of Spamhaus lists and indicate that, if a reflector blocks traffic using blocklists, it very likely uses Spamhaus.

Ultimately the blocklist use and blocking behavior of the reflectors is strongly tied to the organization to which they belong. While inferring the exact organization behind an IP is difficult, we can still explore some high-level organizational aspects of blocklists. We first identify the AS for every reflector, then use the CAIDA AS-to-Organization dataset [8] to map the AS to an organization. Then, we manually partition the orga-

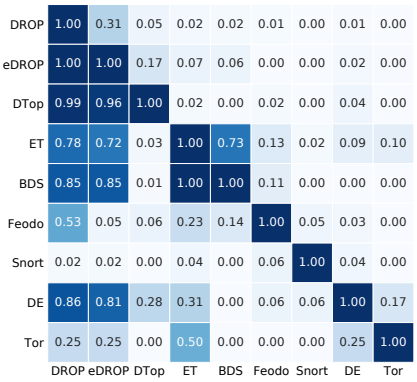

Fig. 4: Pair wise intersection between reflectors that use each blocklist.

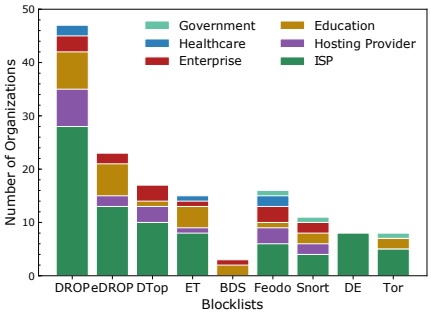

Fig. 5: Breakdown of the number of organizations covered by each blocklist.

nizations into six categories: ISPs (e.g., Comcast), Hosting Providers (e.g., GoDaddy web hosting, AWS cloud computing), Education (e.g., universities), Healthcare (e.g., hospitals), Government (e.g., state and federal agencies), and Enterprise (individual companies owning the IPs).

Figure 5 shows the number of organizations using each blocklist, and their breakdowns by organization category. Most blocklists are used by a wide variety of organizations. Feodo IP Blocklist is the most diverse blocklist in our study, as organizations from all six categories use it. From the perspective of organizations, Educational institutions cover 8 of the 9 blocklists we selected, suggesting a potential preference among universities on using public blocklists.

Validation: Based upon the locations of blocking reflectors, we reached out to two universities that we concluded are using blocklists. In both cases, the blocklists we inferred matched the blocklists they reported using, validating the technique in these two cases. More specifically, University A confirmed our findings that they use BDS IP Ban List, ET Compromised, Spamhaus DROP and Spamhaus EDROP. University B confirmed our findings that they use Spamhaus DROP and Spamhaus EDROP.

5.2 Partial Blocking

Partial blocking is when a reflector blocks some of the blocklist IPs but not all of them. There are many reasons, unrelated to the use of a blocklist, why a reflector may block a blocklist IP. A host may have internal policies that deny access from some network providers, or network administrators may add IPs into their firewall on an ad-hoc basis based on an organization's own policies. These alternate blocking behaviors could overlap with the blocklist IPs we sampled, leading to partial blocking behavior.

Geo-blocking is one cause of partial blocking we identified, where a reflector drops all traffic from a particular country. DShield Top Blocklist, for example, had over 50% of its IPs on January 25, 2020 geo-located in the Netherlands. If a reflector blocks traffic from the Netherlands, then we would observe that the reflector is partially blocking

DShield Top Blocklist. To identify whether a reflector uses geo-blocking, we check whether the reflector *consistently* blocks IPs from a particular country. For all countries related to blocklist IPs we tested, we sample IP addresses from those countries based on four IP location services: MaxMind [24], IP2Location [15], IPDeny [16], and IPIP.net [17], and test against our reflectors. Overall, we identified a small number of reflectors, 614 (0.28%), that consistently block traffic from at least one country.

After removing the geo-blocking reflectors from partial blocking cases, we noticed that a small percentage of reflectors consistently blocked a significant subset of blocklist IPs, but not all, in *every experiment*. This consistency suggests that there is a large overlap between the blocklist and the blocking policy of the reflector. If a reflector blocks over 50% of sampled IPs from a blocklist *every time* we test, we regard the reflector as exhibiting *significant partial blocking* over a blocklist. In total we identified 871 (0.4%) such reflectors. These hosts are probably using a source that is very similar to the blocklists we tested, as previous work has shown that commercial products can aggregate data from public blocklists, and then conduct post-processing to eliminate some content [23]. It is also possible that they are using an older version of the same list, where the content is mostly the same.

Besides these cases, an additional fifth of reflectors demonstrate blocking behavior, as evidenced in Figure 2. Although we do not know the exact reason for the blocking, the result suggests that security-related network blocking is relatively prevalent even among low security hygiene hosts such as these reflectors.

Finally, we had originally hypothesized that network layer blocking would be primarily implemented in border devices (e.g., firewalls, gateways) and thus affect whole network blocks identically. However, when checking reflectors within the same /24s, we find that reflectors under the same /24 frequently do not block the same set of IPs. We refer to this as *inconsistent blocking*. Our experiment found 8,909 (`/24, blocklist`) pairs where multiple reflectors under that /24 block some IPs in that specific blocklist. Among them, 3,263 (36.6%) pairs show inconsistent blocking behavior. This result implies there is considerable intra-network diversity in blocking policy. More analyses and details on the methodology can be found in Chapter 4 in [22].

6 Conclusion

Our paper proposes, implements and tests a technique for inferring the deployment of network-layer blocklists and, further, for attributing the use of particular blocklists by particular hosts. While our technique depends on hosts that are largely quiescent (not sending or receiving much traffic) and use a global increment strategy for IP ID generation (typically those with older operating systems), both of these limitations may be addressable to some extent. Hosts with modest levels of traffic are likely still amenable to testing by using larger sample sizes and more sophisticated statistical testing regimes. As well, while many modern hosts purposely obfuscate the generation of IP ID values, recent work by Klein and Pinkas [19] has demonstrated attacks on these algorithms (in Windows and Linux in particular) which may provide purchase for using the IP ID side channel with more contemporary machines. Future work could leverage these methods to apply our technique to more blocklists with a broader set of reflectors.

Our pilot study covered 220K US hosts, identified blocking behavior in roughly a fourth of all reflectors, but only 2% show clear use of the blocklists we tested against. This difference is puzzling on multiple fronts. It suggests that even among older quiescent hosts that there are significant network security controls in place. Also, it indicates that there may be far more diversity in blocklist usage than we had initially imagined.

References

1. Afroz, S., Tschantz, M.C., Sajid, S., Qazi, S.A., Javed, M., Paxson, V.: Exploring Server-side Blocking of Regions. Tech. rep., ICSI (2018)
2. Anderson, D.: Splinternet Behind the Great Firewall of China. Queue **10**(11), 40–49 (2012)
3. antirez: new tcp scan method. `https://seclists.org/bugtraq/1998/Dec/79`
4. Aryan, S., Aryan, H., Halderman, J.A.: Internet Censorship in Iran: A First Look. In: Proceedings of the 3rd USENIX Workshop on Free and Open Communications on the Internet (FOCI) (2013)
5. Bellovin, S.M.: A Technique for Counting NATted Hosts. In: Proceedings of the 2nd Internet Measurement Conference (IMC). pp. 267–272 (2002)
6. Bhutani, A., Wadhwani, P.: Threat Intelligence Market Size By Component, By Format Type, By Deployment Type, By Application, Industry Analysis Report, Regional Outlook, Growth Potential, Competitive Market Share and Forecast, 2019 – 2025
7. Bouwman, X., Griffioen, H., Egbers, J., Doerr, C., Klievink, B., van Eeten, M.: A Different Cup of TI? The Added Value of Commercial Threat Intelligence. In: Proceedings of the 29th USENIX Security Symposium (USENIX Security). pp. 433–450 (Aug 2020)
8. CAIDA: Inferred AS to Organization Mapping Dataset. `https://www.caida.org/data/as_organizations.xml`
9. Censys – Public Internet Search Engine. `https://censys.io/`
10. Clayton, R., Murdoch, S.J., Watson, R.N.: Ignoring the Great Firewall of China. In: Proceedings of the International Workshop on Privacy Enhancing Technologies (PET). pp. 20–35. Springer (2006)
11. Ensafi, R., Knockel, J., Alexander, G., Crandall, J.R.: Detecting Intentional Packet Drops on the Internet via TCP/IP Side Channels. In: Proceedings of the International Conference on Passive and Active Network Measurement (PAM). pp. 109–118 (2014)
12. FireHOL IP Lists – All Cybercrime IP Feeds. `http://iplists.firehol.org/`
13. Hao, S., Kantchelian, A., Miller, B., Paxson, V., Feamster, N.: PREDATOR: Proactive Recognition and Elimination of Domain Abuse at Time-Of-Registration. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS). pp. 1568–1579. ACM (2016)
14. Hao, S., Thomas, M., Paxson, V., Feamster, N., Kreibich, C., Grier, C., Hollenbeck, S.: Understanding the Domain Registration Behavior of Spammers. In: Proceedings of the ACM Internet Measurement Conference (IMC). pp. 63–76. ACM (2013)
15. IP2Location: IP Address to Identify Geolocation. `https://www.ip2location.com/`
16. IPdeny IP country blocks. `https://www.ipdeny.com/`
17. IPIP.net: The Best IP Geolocation Database. `https://en.ipip.net/`
18. Khattak, S., Fifield, D., Afroz, S., Javed, M., Sundaresan, S., Paxson, V., Murdoch, S.J., McCoy, D.: Do You See What I See? Differential Treatment of Anonymous Users. In: Proceedings of the Network and Distributed System Security Symposium (NDSS) (2016)
19. Klein, A., Pinkas, B.: From IP ID to Device ID and KASLR Bypass. In: Proceedings of the 28th USENIX Security Symposium (USENIX Security). pp. 1063–1080 (2019)

20. Kührer, M., Rossow, C., Holz, T.: Paint It Black: Evaluating the Effectiveness of Malware Blacklists. In: Proceedings of the International Workshop on Recent Advances in Intrusion Detection (RAID). Springer, Springer International Publishing (2014)
21. Lemon, J.: Resisting SYN Flood DoS Attacks with a SYN Cache. In: Proceedings of the BSD Conference (BSDCon). pp. 89–97. USENIX Association, USA (2002)
22. Li, G.: An Empirical Analysis on Threat Intelligence: Data Characteristics and Real-World Uses. Ph.D. thesis, UC San Diego (2020)
23. Li, V.G., Dunn, M., Pearce, P., McCoy, D., Voelker, G.M., Savage, S.: Reading the Tea leaves: A Comparative Analysis of Threat Intelligence. In: Proceedings of the 28th USENIX Security Symposium (USENIX Security). pp. 851–867 (Aug 2019)
24. MaxMind: IP Geolocation and Online Fraud Prevention. `https://www.maxmind.com/`
25. McDonald, A., Bernhard, M., Valenta, L., VanderSloot, B., Scott, W., Sullivan, N., Halderman, J.A., Ensafi, R.: 403 Forbidden: A Global View of CDN Geoblocking. In: Proceedings of the Internet Measurement Conference 2018. pp. 218–230 (2018)
26. NetAcuity. `https://www.digitalelement.com/solutions/`
27. OpenNet Initiative: Survey of Government Internet Filtering Practices Indicates Increasing Internet Censorship (May 2007)
28. Park, J.C., Crandall, J.R.: Empirical Study of a National-Scale Distributed Intrusion Detection System: Backbone-Level Filtering of HTML Responses in China. In: IEEE 30th International Conference on Distributed Computing Systems (ICDCS). pp. 315–326. IEEE (2010)
29. Pearce, P., Ensafi, R., Li, F., Feamster, N., Paxson, V.: Augur: Internet-Wide Detection of Connectivity Disruptions. In: Proceedings of the IEEE Symposium on Security and Privacy (SP). pp. 427–443. IEEE (2017)
30. Pitsillidis, A., Kanich, C., Voelker, G.M., Levchenko, K., Savage, S.: Taster's Choice: A Comparative Analysis of Spam Feeds. In: Proceedings of the ACM Internet Measurement Conference (IMC). pp. 427–440. Boston, MA (Nov 2012)
31. Ponemon Institute LLC: Third Annual Study on Changing Cyber Threat Intelligence: There Has to Be a Better Way (January 2018)
32. Postel, J.: RFC0791: Internet Protocol (1981)
33. Ramachandran, A., Feamster, N., Dagon, D.: Revealing Botnet Membership Using DNSBL Counter-Intelligence. SRUTI **6** (2006)
34. Shackleford, D.: Cyber Threat Intelligence Uses, Successes and Failures: The SANS 2017 CTI Survey. SANS, Tech. Rep. (2017)
35. Sheng, S., Wardman, B., Warner, G., Cranor, L.F., Hong, J., Zhang, C.: An Empirical Analysis of Phishing Blacklists. In: Proceedings of the Conference on Email and Anti-Spam (CEAS) (2009)
36. Singh, R., Nithyanand, R., Afroz, S., Pearce, P., Tschantz, M.C., Gill, P., Paxson, V.: Characterizing the Nature and Dynamics of Tor Exit Blocking. In: Proceedings of the 26th USENIX Security Symposium (USENIX Security). pp. 325–341 (2017)
37. Sinha, S., Bailey, M., Jahanian, F.: Shades of Grey: On the effectiveness of reputation-based "blacklists". In: Proceedings of the 3rd International Conference on Malicious and Unwanted Software (MALWARE). pp. 57–64. IEEE (2008)
38. Spring, N., Mahajan, R., Wetherall, D.: Measuring ISP topologies with Rocketfuel. ACM SIGCOMM Computer Communication Review (CCR) **32**(4), 133–145 (2002)
39. Thomas, K., Amira, R., Ben-Yoash, A., Folger, O., Hardon, A., Berger, A., Bursztein, E., Bailey, M.: The Abuse Sharing Economy: Understanding the Limits of Threat Exchanges. In: Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses. Springer, Springer International Publishing (2016)
40. Tounsi, W., Rais, H.: A survey on technical threat intelligence in the age of sophisticated cyber attacks. Computers & security **72**, 212–233 (2018)

41. University of Oregon Route Views Project. http://www.routeviews.org/routeviews/
42. 2020 Best National University Rankings. https://www.usnews.com/best-colleges/rankings/national-universities (Jan 2020)
43. Zittrain, J., Edelman, B.: Internet Filtering in China. IEEE Internet Computing **7**(2), 70–77 (2003)

A Inference Technique Details

Our technique, while simple in theory, needs to handle real-world scenarios, including packet losses, packet reordering during transition, and other traffic on reflectors. The inference method needs to be *efficient*, *accurate*, and have *low overhead*. Blocklists can change frequently, leaving a short window to infer a stable behavior. As such, for the measurement to finish in a reasonable amount of time requires an efficient inference method. Additionally, the method should also have low false positive and false negative rates so that we can be confident about the result. Finally, it should require as few packets as possible to reduce potential impact on reflectors.

The first step is to find reflectors suitable to our measurement technique. Recall that a suitable reflector should have minimal background traffic, and not be part of a network doing ingress filtering for spoofed packets. To find quiescent hosts, reflectors with low background traffic, we send 24 probes to each candidate host, 1 per second, and repeat the experiment 5 times at different times of the day. We then only select hosts where at least 30% of their IP ID increases are equal to 1 per second — the host did not receive any extra traffic in that one second. We use the 30% threshold to select hosts that are largely "quiet", and thus more likely to yield a perfect signal in the experiment. Next, to identify hosts behind ingress filtering, we acquired 7 vantage points around the world to exercise different paths to the reflector. We sent spoofed packets from our measurement machine to the hosts with spoofed source addresses corresponding to the 7 vantage points, and then collected responses at each vantage point. We only select the hosts that send responses to all 7 vantage points, meaning they did not drop spoofed packets on any of the exercised network paths.

Next, we describe how we infer if a given reflector blocks an IP using multiple *trials*. We define a *trial* as a single experiment that tests if a reflector blocks one blocklist IP. Figure 6 shows the process of one trial. For each trial, the measurement machine sends five consecutive *probe packets* to the reflector, with each packet being sent one second apart. In our experiment, the probe packets are TCP SYN-ACK packets and we get IP IDs from response RST packets. Between the third and fourth probe packets, the measurement machine sends five *spoofed packets*, also TCP SYN-ACK, with source IPs equal to the blocklist IP. And between the fourth and the fifth probe packets, it sends another five spoofed packets. We send the five spoofed packets 0.15 seconds apart consecutively each time, spreading them across the one-second window between two probes.

We then inspect the increases between the IP IDs in the packets received by the measurement machine. Ideally, assuming no additional traffic and no packet loss, the IP ID should increase by exactly one between consecutive probes. For the last two deltas, since we send the spoofed packets in between our probe packets, the final IP ID increases will be different based on the host's blocking behavior.

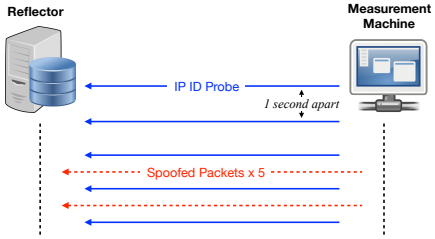

Fig. 6: Blocking inference methodology. Solid blue lines are *probe packets*, dashed red lines are *spoofed packets*.

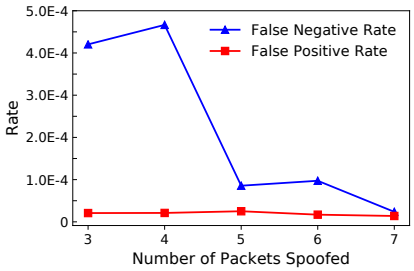

Fig. 8: Experiment design and false positive and false negative analysis

If the reflector does not block the blocklist IP, then we will observe an IP ID increase sequence in our received RST responses that is: [+1, +1, +6, +6]. Here the last two deltas are +6 since the reflector does not block the blocklist IP and thus responds to spoofed packets, causing IP ID to increase by 5, and our probe packet causes it to increase by another 1, which together make +6.

On the other hand, if the reflector blocks the blocklist IP, then we will see an IP ID increase sequence that is: [+1, +1, +1, +1]. Here the last two deltas are +1 since the reflector blocks the blocklist IP, leading to no extra change in IP ID.

The first three probes—corresponding to the first two IP ID deltas—act as a control. The last two "probe and spoof" patterns perform the actual experiment. Seeing the initial two "+1" indicates this host is in a quiet period (no extra network traffic). Therefore, we can be more confident that the following IP ID jump ("+6" in our case) is because of our experiment. While the choice of the numbers in the experiment may seem arbitrary, there is a rationale behind the choice which we will discuss in following sections.

A.1 Inference Criteria

We now look at the criteria to infer if a reflector blocks a blocklist IP or not. Our limited vantage point from the measurement machine limits our information to the IP IDs seen from the reflector. Moreover, we desire to be conservative when inferring blocking. Thus, our approach is to try the same trial, between a reflector and a blocklist IP, until we get a "perfect signal"—a response which matches all the criteria below:

1. The measurement machine received exactly five RST responses from the reflector.
2. The five responses are received one second apart consecutively.
3. The IP ID increase sequence is either [+1, +1, +6, +6], which we will conclude as no blocking, or [+1, +1, +1, +1], which we will conclude as blocking.
4. If any of the above three criteria are not met, we repeat the same experiment again. We repeat up to 15 trials before giving up.

The first requirement ensures no packet loss. The second requirement ensures responses we received reflect the real IP ID changes in the reflector. The Internet does

not guarantee the order of packet arrival. Although we send one probe packet per second, these packets might not arrive at the reflector in the same order. Thus, the IP ID sequence from the response packets might not represent the real order of IP ID changes at the host. Hence, by requiring that the response packets cannot be less than 0.85 or more than 1.15 seconds apart we can minimize the probability of reordered packets.

The third requirement is the core of our inference logic. Since we ignore everything other than an IP ID increase sequence of [+1, +1, +1, +1] or [+1, +1, +6, +6], we can assure that our inference of blocking is conservative. If we saw a sequence of [+1, +1, +1, +1] but the reflector does not block the blocklist IP, that would mean all 10 spoofed packets were lost. On the other hand, if we see [+1, +1, +6, +6] and the reflector actually blocks the blocklist IP, that would mean there are exactly five extra packets generated by the reflector during each of the last two seconds. Both cases are very unlikely, which we will demonstrate next with an analysis of false positives and false negatives.

A.2 False Positive and False Negative Analysis

For our experiment, a *false positive* is when a reflector is not blocking a blocklist IP, but we mistakenly conclude it is blocking. On the other hand, a *false negative* is when a reflector is blocking a blocklist IP, but we mistakenly conclude it is not. To evaluate false positive and false negative rates, we conduct experiments on *all* the reflectors under consideration and measure the false positive and false negative rates.

For false positive evaluation, we first acquire a list of IPs that are verifiably not being blocked by reflectors. Since we own these IPs, we can easily verify by directly probing reflectors from these IPs. We acquired and tested 1,265 IPs from five different /24s. Then we probe reflectors and send the spoofed packets with source addresses set to these pre-selected IPs. Since these IPs are not being blocked, if we observe an IP ID increase sequence of [+1, +1, +1, +1], then we know it is a false positive.

For false negatives, we run the experiment with only probe packets, and no spoofed packets. This scenario is equivalent to the one where the reflector blocks the spoofed IP. If we observe an IP ID increase sequence of [+1, +1, +6, +6], then we know it was due to the background traffic at the reflector and hence is a false negative.

Although we present the experiment design with five spoofed packets in each of the last two seconds, we also experimented with a range of numbers and calculated their false positive and negative rates. We tested 15 times with spoofed packets equal to 3, 4, 5, 6, and 7 with every reflector, and we repeated the experiment again on a different day. The final results are shown in Figure 7.

We need to trade off between keeping false positive and negative rates low while generating as little traffic as possible. We choose 5 spoofed packets as a balance. By sending 5 spoofed packets, we get a false positive rate of 2.5e-5, and a false negative rate of 8.5e-5. Furthermore, we also experimented with strategies where we send 4 probe packets, from which we get 3 IP ID deltas, and sending 6 probe packets, from which we get 5 IP ID deltas. With only 3 deltas we suffer a higher false negative rate, as it is easier for the reflector to show the same IP ID increase sequence with extra traffic. With 6 probes, on the other hand, we prolong the experiment, making it harder to get a "perfect signal". Thus, our choice of 5 probe packets with 5 spoofed packets in between is a good balance between competing factors.