Blacklists Assemble: Aggregating Blacklists for Accuracy

Sivaramakrishnan Ramanthan[1], Jelena Mirkovic[2], and Minlan Yu[3]

[1]University of Southern California

[2]University of Southern California/Information Sciences Institute

[3]Harvard University

ABSTRACT

IP address blacklists are a useful defense against various cyberattacks. Because they contain IP addresses of known offenders, they can be used to preventively filter unwanted traffic, and reduce the load on more resource intensive defenses. Yet, blacklists today suffer from several drawbacks. First, they are compiled and updated using proprietary methods, and thus it is hard to evaluate accuracy and freshness of their information. Second, blacklists often focus on a single attack type, e.g., spam, while compromised machines are constantly and indiscriminately reused for many attacks. Finally, blacklists contain IP addresses, which lowers their accuracy in networks that use dynamic addressing.

We propose BLAG, a sophisticated approach to select, aggregate and selectively expand only the accurate pieces of information from multiple blacklists. BLAG calculates information about accuracy of each blacklist over regions of address space, and uses recommendation systems to select most reputable and accurate pieces of information to aggregate into its master blacklist. This aggregation increases recall by 3–14%, compared to the best-performing blacklist, while preserving high specificity. After aggregation, BLAG identifies networks that have dynamic addressing or a high degree of mismanagement. IP addresses from such networks are selectively expanded into /24 prefixes. This further increases offender detection by 293–411%, with minimal loss in specificity. Overall, BLAG achieves high specificity 85–89% and high recall 26–61%, which makes it a promising approach for blacklist generation.

1. INTRODUCTION

Compromised devices are constantly being drafted into botnets and misused for attacks, such as sending spam and phishing emails [71], scanning for vulnerabilities, participating in denial-of-service attacks [27, 32, 84], and spreading malware [79]. *IP blacklists* ("blacklists" for short), which contain identities of prior known offenders, can be helpful as the first-layer defense. Assuming that prior offenders are likely to reoffend, filtering traffic from blacklisted sources can proactively prevent recurrent attacks. It also helps during high-volume attacks, such as denial of service or worm spread, because it reduces load on more resource-intensive defenses, such as network intrusion detection systems [68] and DDoS scrubbers [13]. Blacklists are widely used by network providers [45] and researchers [67, 72, 75, 83], but they have several drawbacks, which we seek to address in this paper.

Blacklists are created by organizations, which monitor some regions of the Internet for specific malicious activities. This limited observation introduces two deficiencies. First, blacklists are often attack-type-specific and will miss offenders who engage in a different malicious activity (e.g., a spam blacklist will contain known spammers but not known booters). On the other hand, compromised hosts are traded on black market and reused for many malicious activities [53, 79, 81, 87], indiscriminately and consistently. A host, which sends spam today could engage in DDoS or spread ransomware tomorrow. Thus, it would make sense to create generic blacklists, which aggregate information from attack-type-specific lists, to increase offender detection. Second, blacklists accuracy may vary a lot. A blacklist may miss certain attacks, because they occur in parts of the Internet that the blacklist's maintainer cannot observe (e.g., a US-based spam blacklist, created by analyzing e-mails on a large mail server, may miss spam attacks launched by Chinese hosts on Brazilian users). A blacklist may also miss observable attacks, or falsely list legitimate addresses, depending on the tuning of its detection algorithm. Thus, each blacklist will have portions of accurate information, which we would want to include in aggregation, and portions of inaccurate information, which we would want to exclude. To achieve such selective aggregation, we need a way to identify regions of the Internet address space where a given blacklist performs well or poorly. Third, blacklists may miss offenders or falsely filter legitimate traffic due to dynamic addressing [56, 64, 67]. Blacklists are also reactive, and will miss new offenders from networks that have historically harbored offenders in the past [87]. We would like to identify such dynamic and mismanaged networks, where we can replace blacklisted addresses with prefixes to improve offender detection. We would also like to perform such expansion *selectively*, i.e., only when it does not lead to large increase in false positives.

In this paper we propose BLAG, a sophisticated blacklist aggregation approach, which addresses the problems we outlined. BLAG aims to increase recall (rate of offender identification or true positives) over individual blacklists, while maintaining high specificity (low rate of false identification or false positives). BLAG has three novel contributions, compared to prior work on blacklist aggregation [77, 83].

1. It uses an estimate of false positives (specificity) over different blacklists and IP-address regions to identify areas of blacklists to aggregate. This increases specificity of the aggregated master blacklist by 3.9–10.9%, when compared with naive aggregation of all blacklists.

2. It uses recommendation systems to overcome information sparsity problem in blacklists. This helps BLAG rely on good reputation of some blacklists to boost weak signals and increase recall. BLAG improves offender detection accuracy by 3.3–14.3% over individual blacklists.

3. It evaluates each region of Internet address space for dynamic addressing or mismanagement. If either is found, BLAG may replace the addresses from the region with /24 prefixes, if anticipated loss of specificity is acceptable. This *selective expansion* further improves recall, by 411% with a maximum of 14% loss in specificity.

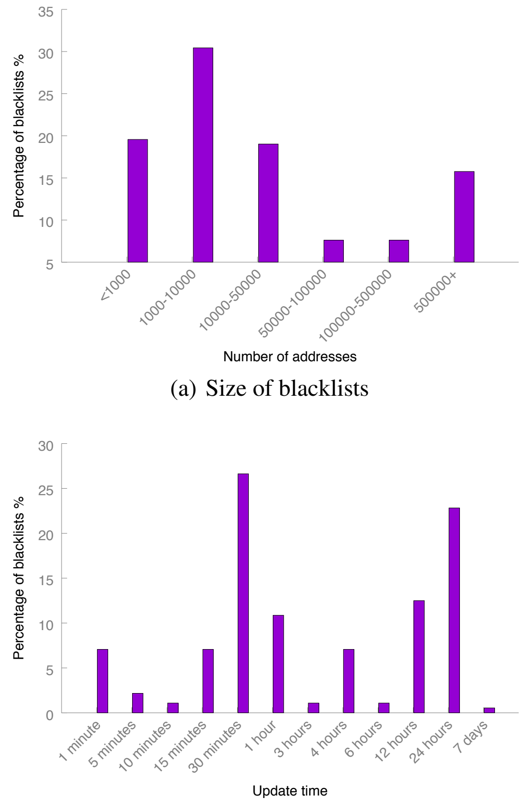
BLAG outperforms naive aggregation of all monitored blacklists, and achieves higher specificity while increasing recall by 226%. BLAG also outperforms PRESTA [83], a recently proposed blacklist aggregation approach by achieving 107–402% higher specificity than PRESTA.

2. DATASETS, METRICS AND USE CASES

To illustrate the problems experienced by the current blacklists, we have analyzed 157 publicly available blacklists, collected regularly over a one-year period. We have further collected several ground-truth datasets containing known-legitimate and known-malicious traffic sources. We use these datasets to evaluate performance of current blacklists and to evaluate BLAG. We describe our datasets and metrics in this section, and discuss performance goals and blacklist use cases.

2.1 Datasets

Black. This dataset provides input into BLAG and competing approaches. We have collected 157 publicly available blacklists continuously for 13 months starting from January 2016 to February 2017. Each blacklist may be updated at a different frequency by its provider, ranging from 15 minutes to 7 days. The distribution of update times for our dataset is show in Figure 1(b). Our collection algorithm detects the update frequency for each blacklist and refreshes its snapshot regularly. We have collected around 176 million blacklisted addresses over 23,483 autonomous systems. Our blacklist dataset is representative of different attack vectors such as spam, malware, DDoS attacks, ransomeware, etc. Table 1 shows the blacklists maintainers roughly classified into four categories based on the attack type they monitor and the number of blacklists maintained by them. Our dataset includes popular blacklists such as DShield [59], Nixspam [66], Spamhaus [44], Alienvault [1], Project Honeypot [38], Abuse.ch [48], Emerging Threats [18] and malc0de [30]. Figure 1(a) shows the blacklist size distribution in the dataset. On one hand, we have large blacklists (15.76%) listing more than 500,000 addresses and on the other, we have small blacklists (19.56%) which list less than 1,000 addresses.



(a) Size of blacklists

(b) Update times of blacklists

Figure 1: Blacklist size and update times.

Ground truth: Mailxam (Mailinator+Alexa+Ham). This dataset contains one source of malicious addresses and two sources of legitimate addresses, collected over the same month of June, 2016. Simultaneous collection is important, because an address may be malicious at one time, and cleaned afterwards. Our malicious source comes from **Mailinator** [29], a service, which allows users to redirect unwanted e-mails to a public inbox. We filter e-mails from these public inboxes during June 2016, using Spam Asssassin [65] to obtain around 2.3 M spam e-mails, sent by around 393 K addresses. These addresses form our malicious dataset. Our first source of legitimate addresses are **Alexa**'s [26] top 500 K websites

Type	#	Blacklist Maintainers
Malware	51	Emerging threats [19], Malware Bytes [21], Clean MX [11], Jigsaw security [25], CyberCrime [15], Swiss security blog [48], Bambenek [5], NoThink [35], I-Blocklist [22], NoVirusThanks [36], DYN [23], Malc0de [30], Malware domain list [31], Cyber Threat Alliance [14], Botscout [51], ASProx Tracker [2]
Reputation	49	Emerging threats [19], Graphiclineweb [24], Alienvault [1], Binary Defense Systems [6], CINSscore [9], Swiss Security Blog [48], Blocklist.de [7], I-Blocklist [22], Cisco Talos [10], Bad IPs [4], Blocklist Project [52]
Spam	48	Spamhaus drop and edrop [44], Stop Forum Spam [47], Chaosreigns [3], Lashback [28], Nixspam [66], Project Honeypot [38], Sblam! [41], Turris [20], Malware bytes [21], Cleantalk [12], My IP [33], Pushing inertia [39], BadIPs [4]
Attacks	36	I-Blocklist [22], Malware Bytes [21], Snort Labs [43], Jigsaw Security [25], TrustedSec [49], Haley [8], Darklist [17], SIP blacklist [46], VoIPBL [50], DShield [59], NoThink [35], OpenBL [37], Cruzit [42], BruteforceBlocker [16], Clean MX [11], Bad IPs [4], MaxMind [40]

Table 1: Four types of blacklists, roughly categorized by the type of malicious activities they capture. Each row gives the number of blacklists and blacklist maintainers for that type.

Dataset	Start time	Type	Sources
Mailxam	06/01/2016	Malicious	Mailinator (393 K)
30 days		Legitimate	Alexa (284 K)
		Legitimate	Ham (45 K)
Miraixa	09/01/2016	Malicious	Mirai (232 K)
31 days		Legitimate	Alexa (330K)
Darkexa	02/01/2017	Malicious	Darknet (3.9 M)
16 days		Legitimate	Alexa (281K)

Table 2: Ground-truth datasets used in this study, collected in 2016/2017.

mined in June 2016. Out of this set we remove websites that may host malware, using Google Safe Browsing API [60] for detection. Afterwards, we convert the domain names into addresses using DNS, which leaves around 284 K addresses. Our second source of legitimate addresses, **Ham**, comes from our human user study. This study was reviewed and approved by our IRB. We recruited 37 volunteers, who allowed us automated access to their GMail inbox, during June 2016. We scanned each participant's GMail account using a plugin, which we developed. Our plugin used OAuth2 protocol to access GMail without requiring the participant's GMail credentials, and it used regular expressions to extract a sender's address, time and label for each e-mail. The label in GMail can be assigned by a user or by GMail and it is usually "spam", "inbox" or a user-defined label like "conference". We harvested information only from e-mails that have labels other than "spam". Our scanning generates as output a list of {sender IP address, time} tuples, which we save. We collected no identifying information about our study participants, thus this collection posed no privacy risk. We extracted around 178 K e-mail records, sent by around 45 K addresses.

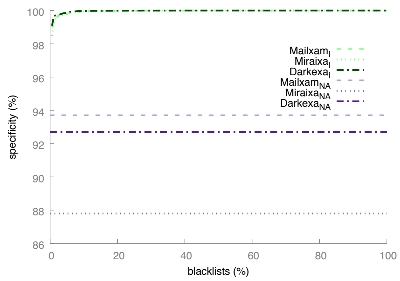
Ground truth: Miraixa (Mirai+Alexa). This dataset contains one source of malicious and one of legitimate addresses, both collected during September 2016. Our malicious source comes from Netlab's [34] scans of **Mirai**-infected hosts during September 2016. There were around 232 K infected hosts. Our legitimate source comes from **Alexa**'s top 500 K websites, mined in September 2016, and filtered as described for the previous dataset. We collected around 330 K legitimate addresses.

Ground truth: Darkexa (Darknet+Alexa). This dataset contains one source of malicious and one of legitimate addresses, both collected in February 2017. The malicious source comes from sources of TCP scans (SYN packets) sent to CAIDA's /8 **Darknet** [57] in February 2017. These sources may be spoofed, but we have no way to identify and remove spoofed scans. We collected around 3.9 M malicious addresses. Our legitimate source comes from **Alexa**'s top 500 K websites, mined in February 2017 and filtered as described for previous datasets. We collected around 281 K legitimate addresses.

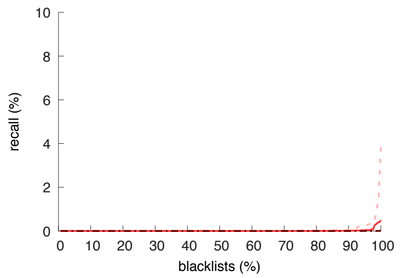
Our three datasets contain sources of diverse attacks: spam, DDoS or vulnerability scans. This allows us to test how well BLAG could prevent these attacks if used to filter traffic to a deploying network.

Limitations. Our datasets suffer from several limitations. The Black dataset contains only the publicly available blacklists, while many providers also offer for-pay blacklists that may be larger and more accurate. We chose to use only the public blacklists because: (1) these blacklists are widely used and we wanted to evaluate BLAG's benefits for an average user, (2) we wanted our work to be repeatable, and using public blacklists enables us to freely share our data. We plan to share all the datasets from this section and our BLAG blacklist by the final version deadline. We believe that BLAG's benefits would carry over to for-pay datasets, because its mechanism is generic.

Another limitation is that legitimate and malicious datasets are small and imperfect. They only capture a sample of legitimate/malicious addresses that were active in the Internet at a given point in time and for a given legitimate or



(a) Specificity



(b) Recall

Figure 2: Blacklist specificity is usually high but recall is low. Naive aggregation (denoted by NA) has higher recall, but with lower specificity.

malicious purpose. We also rely on other security technologies, such as Google safe browsing or SpamAssasin to label an address as legitimate or malicious at a given point in time. These limitations are present in other published works [58, 62, 70, 77, 83, 86], which use similarly-sized malicious and legitimate datasets, and rely on secondary technology, as do we, to establish maliciousness at a given time. These limitations cannot be avoided, as there is no complete, 100% accurate list of legitimate and malicious addresses in the Internet nor in any specific network, at any given point in time.

Our datasets contain addresses of different types. For example, in Miraixa, malicious addresses belong to IoT devices, while legitimate addresses belong to Web servers. This is not a limitation. Internet traffic contains a variety of hosts, which engage in a variety of behaviors. Our datasets contain a small subset of these hosts and behaviors, where we could establish legitimacy or maliciousness, with some degree of confidence. They are not perfect, but we hope they are sufficient to provide a common ground truth to evaluate BLAG and competing approaches.

2.2 Metrics

We measure performance of blacklists using *recall* and *specificity*. Recall measures the percentage of offenders (out of some ground-truth set) that were blacklisted. Specificity measures the percentage of legitimate hosts (out of some ground-truth set) that were not blacklisted.

2.3 Use Cases and Performance Goals

Performance goals for any given blacklist depend on when and how it is used. If a blacklist is used *preventively*, as a first-layer defense, and is on all the time, it is very important that the blacklist has high specificity. This is to ensure that no legitimate traffic is regularly dropped. Current blacklists have high specificity (near 100%) but extremely low recall. If a blacklist is used *reactively*, to prioritize traffic drops during a high volume attack, such as DDoS or a worm infection, it is very important to maximize recall, and achieve some reasonably high specificity. Our work targets this second use case. We will show in Section 5 that BLAG achieves more than 86% specificity, while significantly increasing recall, compared to individual blacklists.

3. PROBLEMS WITH CURRENT BLACKLISTS

In this section we illustrate the problems that blacklists have and that we aim to handle. To estimate specificity and recall for individual blacklists, we calculate the *daily* overlap between addresses reported in the malicious and legitimate sources for Mailxam, Miraixa and Darkexa datasets and the addresses reported by the individual blacklists. We then report the total percentage of legitimate (specificity) and malicious (recall) addresses listed over the course of the entire ground-truth dataset.

We first show that all blacklists have generally high specificity but poor recall. This motivates the need for their aggregation. We further show that naive aggregation fails. It increases recall but severely lowers specificity. We show that this occurs because each blacklist's specificity varies a lot spatially, i.e., over regions of IPv4 address space. This motivates us to identify and aggregate only those portions of blacklists that have high specificity. [Sivaram: Finally, we show the reactive nature of blacklists – that is blacklists list attackers only after a attack has been listed. To further improve attack detection, we see that expanding addresses into its corresponding prefix can potentially increase the number of attackers detected. This is warranted by the study that mismanaged networks are prone to host more attackers than well managed networks. However, we also find that expanding all addresses into prefixes further reduces the specificity.]

3.1 Missed Attacks

Individual blacklists have low recall as shown in Figure 2(b), for our three datasets. The best recall a blacklist from our Black dataset had was 13.5% on Mailxam, 5.6% on Miraixa and 8.4% on Darkexa. About x,y,z% of blacklists do not even report a single attacker. Previous blacklist studies have similarly observed high specificity and low recall [76][Sivaram: cite paint it black]. A key reason for low recall is that blacklists monitor only on specific type of attack. However, compromised machines are constantly drafted into botnets for initiating various types of attacks. A compromised machine used for generating spam emails one day, can be used for generating DDoS attack a few months later.

To improve attack detection, we could combine blacklists of different attack types to detect more attackers. Also, blacklists are rich in historical data which could be further used to improve recall by listing potential re-offenders. One approach is to include every address ever seen on any blacklist into a *historical blacklist*. We show how this approach of combining all blacklists of different attack type including historical data would perform on our ground-truth datasets with regard to specificity (Figure 2(a)) with a horizontal bar. Historical blacklists has higher recall than any monitored blacklist. About 18.7%, 13.3% and 20.5% of malicious addresses from Mailxam, Miraixa and Darkexa datasets are detected for naive aggregation, clearly indicating that blacklists of different attack type over time is useful in uncovering more attackers.

Implications: Blacklists generally have low recall. Therefore, we aggregating blacklists of different attack type over time can increase recall.

3.2 Varying Specificity

Specificity is generally high ($> 94.2\%$) for individual blacklists, which means that no individual blacklist will erroneously list many legitimate sources. We see that 94% of blacklists have 100% specificity and the poorest performing blacklist has a specificity of 93%. [Sivaram: Also write for other data.]

To improve recall, we suggested to aggregation of all blacklists over time. However, this can decrease specificity. Figure shows the drop in specificity of the combined blacklists of different attack type. The specificity drops by x,y,z% for the three dataset. This can occur due to two reasons. First, Blacklist maintainer uses their own proprietary algorithm to include or exclude an address from a blacklist. This can have certain amount of false positives. Second, although historical blacklist data has shown to improve recall, there are many addresses which are no longer malicious. Therefore, historical blacklists, can further amplify the number of false positives in the combined blacklists. The challenge we address in this paper is how to design a smarter aggregation approach, which achieves better recall with minimal loss in specificity.

Implications: Blacklist accuracy varies due to maintainers propriety algorithms and historical data can contain many addresses which are no longer malicious. Therefore, we aim to devise an approach that can identify areas where a blacklist are very accurate and include listings only from those areas in aggregation.

3.3 Address volatility

Blacklists list IP addresses. But in networks that use dynamic addressing, an offender's address can change over time. There are many dynamically-addressed networks today which may affect the performance of blacklists. Thomas et al. [80] observed that devices using Google services are assigned an average of 20 addresses over two weeks. Dynamic nature of addresses was also observed in [56, 64, 67, 73], which estimate dynamic addressing to be prevalent in 8–20% of the Internet.

Another problem with listing addresses is that blacklists can only be reactive, that is, they catch only previously known offenders. Zhang et al. [87] showed the correlation between network mismanagement and maliciousness – malicious entities are often concentrated in few mismanaged networks. If we could identify such mismanaged networks, blacklists could become proactive in that space, by listing the entire network as soon as a few offenders are detected. We observe a pattern of frequent reoffense from several networks in our Black dataset, which is aligned with findings of Zhang et al. About 99.6% of blacklisted addresses reside in the same /16 prefix, and 82.3% of blacklisted addresses reside in the same /24 prefix, as another blacklisted address.

We could potentially expand addresses into prefixes. In Figure we see that by expanding any listed address across historical blacklist data, we see that the recall increases by. However expanding every address into prefix can amplify the number of legitimate addresses even further. Figure shows the specificity after expanding every address into prefix.

Implications: Detecting potential dynamic addressing and mismanagement in networks can provide indications where coarser blacklisting can lead to improved recall. However, such addresses such be selectively expanded to maintain high specificity.

4. BLAG DESIGN

[Jelena: speak somewhere that you use /16 in recommendation system] In this section we present the design of our system – BLAG, which selects accurate information from blacklists and aggregates it. We illustrate the system in Figure 3. We assume some given network wants to deploy BLAG for blacklist aggregation. BLAG starts with a set of recently obtained blacklists (B), and prior blacklist observations (P). Also, BLAG uses some set of known-legitimate senders (L) that generated recent traffic to the deploying network. This set is necessary to estimate specificity of various blacklists from B, and select which portions to aggregate. A university network could form this set, for example, by choosing sender addresses from non-spam messages, addresses from other colleges and universities, addresses of popular Web and DNS servers, etc. BLAG selects some addresses from (B) and (P) to be included in its aggregate master blacklist. Periodically, BLAG updates (P) with information from (B), refreshes (B) with updated listings, refreshes (L) with current known-legitimate senders and recalculates aggregated master blacklist. This process repeats indefinitely.

Our goals for BLAG include:

1. *Evaluate* the quality of each blacklist's information for a given address space. We achieve this by calculating a reputation score for each address and each blacklist, and for each /16 network listed by that blacklist. This score is a function of the blacklist's accuracy in report-

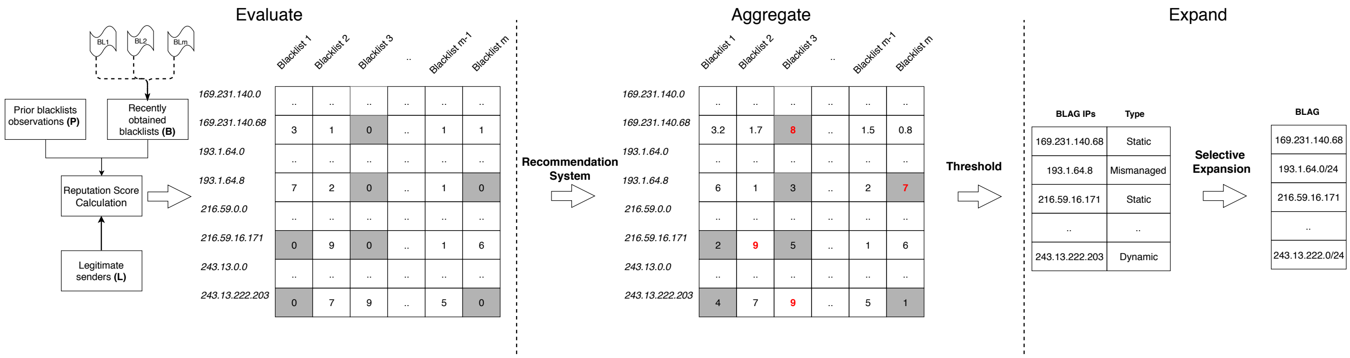


Figure 3: BLAG implementation consists of assigning reputation scores to addresses from different blacklists. Then, recommender system generates scores for addresses, which do not have a score in a given blacklist (shaded blocks). Addresses that have at least one score greater than the α threshold (red numbers) are used for aggregation. These addresses will be put on the master blacklist. Finally, we selectively expand addresses from dynamic or mismanaged networks into /24 prefixes, if we project that this expansion will not severely lower specificity.

ing prior offenses from the same address space region, i.e., it is an estimate of the expected gain in recall and specificity, if this portion of the blacklist were included in aggregation. Similar to prior work in PRESTA [83], we also take into account the listing's age, and favor inclusion of offenders that were recently listed. Thus, the reputation score is also the function of the address's history of offense. We describe score calculation in Section 4.1.

2. *Aggregate* high-quality pieces of information into the master blacklist. We use recommender systems to calculate missing reputation scores, i.e., to predict the likelihood of re-offense of an address within or across blacklists. We then use a threshold-based approach on these reputation scores to filter out unreliable information. This process is explained in Section 4.2.

3. *Expand* some addresses into prefixes on the master blacklist to increase recall. We expand those addresses that we believe are dynamically allocated, or addresses that belong to mismanaged prefixes. Our expansion method is also selective – it tries to balance the gain in recall against the loss in specificity, and is explained in Section 4.3.

4.1 Reputation Scores: Evaluating Quality

BLAG starts its aggregation by first generating a reputation score for each address a and blacklist b. Our reputation score for this listing is a sum of two scores: the *historical offense score* $ho_{a,b}$ and the *safety score* $s_{a,b}$.

$$r_{a,b} = \frac{ho_{a,b} + s_{a,b}}{2} \tag{1}$$

The score ranges from 0 to 1. During the aggregation phase, we multiply the scores by 10, to speed up the convergence of the recommendation system. A higher reputation score indicates that the listed address has a higher probability of re-offense, and the blacklist listing the address has high specificity in the given address region.

Address's history of offense $ho_{a,b}$:
Historical blacklist data can be a valuable source to detect potential re-offenders, and previous studies have shown recent offenders are more likely to re-offend [83]. We define historical offense score $ho_{a,b}$ as:

$$ho_{a,b} = \frac{1}{2^{\frac{t - t_{out}}{l}}} \tag{2}$$

where l is a constant, which we set empirically (discussed in Section 6.3), t_{out} is the de-listing (removal) time of a at blacklist b and t is the time when the score is calculated. The exponential factor ensures that the score decays exponentially over time, giving higher weight to recent offenders. If the address a is currently listed in b, we set its historical offense score to 1.

Blacklist's safety $s_{a,b}$: In Section 3.2, we illustrated how specificity of a given blacklist varies over the Internet's address space. We capture this dependency in the safety score, with higher values denoting higher estimate of specificity. Let $net(a)$ be a /24 prefix containing address a and $o_{net(a)}$ be the set of all addresses from $net(a)$ that were ever reported in blacklist b. Let e be the set of all listed addresses that are also in (L). We define safety of blacklist b for address a as:

$$s_{a,b} = 1 - \frac{|e \cap o_{net(a)}|}{|o_{net(a)}|} = 1 - \frac{F_a}{|o_{net(a)}|} \tag{3}$$

i.e., the safety of a blacklist is the fraction of the addresses it reports within $net(a)$ that are not misclassifications and F_a represents the number of misclassified addresses in $net(a)$.

4.2 Recommender System: Calculating Missing Scores

After all the reputation scores are calculated, BLAG places them into a *score matrix* where blacklists are at the columns

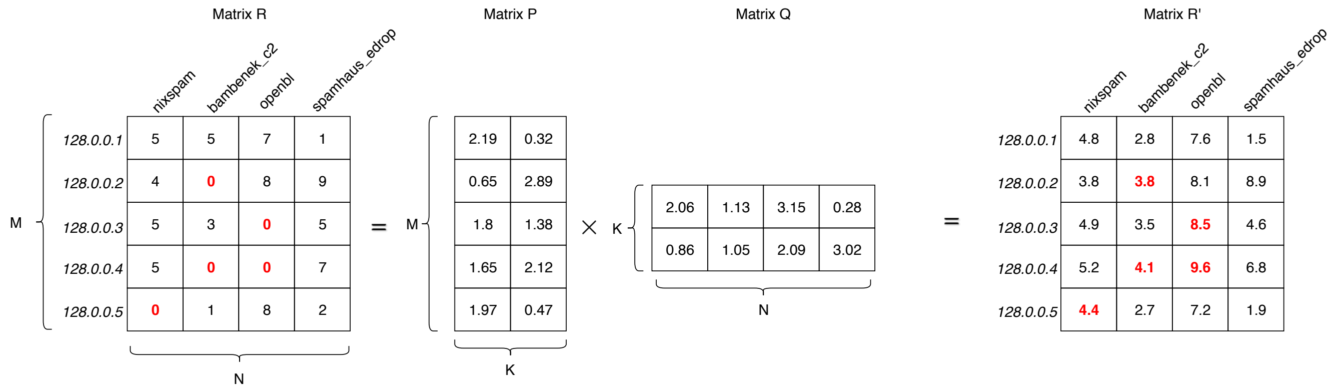


Figure 4: Latent factorization of the score matrix R, a $M \times N$ matrix, where M is the number of addresses and N is the number of blacklists. The cells indicate reputation scores. Addresses not listed in a given blacklist are assigned a zero score. Score matrix is factorized into two matrices of $M \times K$ and $K \times N$, and the cross product results in a new matrix R', which updates the zero score cells with a positive value.

and listed addresses at the rows as shown in Figure 4. BLAG creates a score matrix for every /16 prefix in the Black dataset. Each cell in the score matrix holds the reputation score $r_{a,b}$ for the given row (address a) and given column (blacklist b). This matrix is very sparse. We fill the empty cells by using a recommendation system. This helps us, in some cases, to elevate low-score addresses on some blacklists, into high-score addresses on other, more reputable blacklists. Such addresses then propagate to the aggregated master blacklist, and help us improve recall without a great loss in specificity.

Recommendation systems are usually used to predict future product ratings by some users, given a set of past ratings of same or related products, by target users and other similar users. A well-known example is the Netflix recommendation system [61], which may recommend a new movie M to user U by relying on the U's past ratings of movies similar to M, and on ratings that users similar to U have given to M or movies similar to M. In our context, addresses are the products that are being evaluated, and blacklists are users assigning the rating. We view the reputation score as the rating.

Two most commonly used recommendation systems are a content-based recommendation system [69] and collaborative filtering [74]. A content-based recommendation system requires explicit definition of features describing the relationship between blacklists and addresses. Such features are hard to obtain, because each blacklist uses proprietary algorithms and private observations to decide when to list an address. We instead use *collaborative filtering*, as it infers information about the relationship between a blacklist and an address using only the existing reputation scores.

Figure 4 illustrates the recommendation system's operation. Let M and N represent the set of addresses and blacklists, respectively. Let R be a score matrix of size $|MxN|$ which consists of reputation scores quantifying the maliciousness of an address being listed by a given blacklist. For example in Figure 4 score matrix R consists of four blacklists ($M = 4$), and five addresses ($N = 5$). Every address need not be present in every blacklist, which makes score matrix R sparse. Address 128.0.0.1 listed in *nixspam* blacklist has a reputation score of 5 (on the scale from 0 to 10). Address 128.0.0.4 has a score of zero in *openbl* blacklist, where it has never been listed. There are latent (unknown) features of blacklists and addresses that lead to an address being listed. Let the number of latent features that influence reputation scores of addresses in blacklists be K (see Section 6.4 for how we choose the value of K). Our goal is to estimate the unknown scores in the sparse score matrix R by estimating two matrices $P(|MxK|)$ and $Q(|NxK|)$, which are factors of score matrix R, such that their cross product is approximately equal to known values in R. In other words, matrix factorization is used on R to obtain factor matrices P and Q such that:

$$R \approx P \times Q^T = R' \tag{4}$$

We obtain the values of latent matrices P and Q using gradient descent [63], which randomly assigns values to P and Q and estimates how different the product of P and Q is from the original score matrix R. We use root mean squared error (RMSE) to estimate the difference. Gradient descent tries to minimize RMSE iteratively. We discuss in Section 6.4 the number of iterations required to have a small RMSE.

After obtaining matrices P and Q, each row in P represents the association strength between addresses and latent features K, and each row in Q represents the association strength between blacklists and latent features K. To obtain an unknown reputation score for an address a and blacklist b, the dot product of two latent vectors corresponding to address a and blacklist b is calculated as follows:

$$r_{\mathrm{a,b}} = p_a^T q_b \tag{5}$$

Where p_a defines the association strength of address a with features K and q_b defines the association strength of blacklist b with features K. Consider addresses 128.0.0.3 and 128.0.0.4 in Figure 4, which are not listed in the *openbl* blacklist. Both of these addresses have similar reputation scores in other blacklists (with *nixspam*'s scores of 5 and 5, and *spamhaus_edrop*'s scores of 5 and 7). Intuitively, if these addresses where to be listed in the *openbl* blacklist, we can expect them to have similar scores. Recommendation system captures this pattern. Also *openbl* tends to have a little higher scores for the addresses it lists, compared to other blacklists. This regularity is also captured by the recommendation system. The system generates the R' score matrix, where scores of 8.5 and 9.6 are assigned to addresses 128.0.0.3 and 128.0.0.4 respectively, for the *openbl* blacklist.

After we have calculated all the missing scores, and filled in the empty cells in score matrix R, we proceed to construct the *aggregated master blacklist*. To generate the master blacklist, we observe all blacklists $B = \{b_1, b_2, ..., b_n\}$ and then use a threshold α (choice of α values is discussed in Section 6.5) to include all the addresses a for which the following holds : $\exists b \in B | r_{a,b} \geq \alpha$.

4.3 Selective Expansion: From Addresses to Prefixes

We have discussed in Section 3.3 why it would be useful to identify and expand addresses in dynamic and mismanaged networks into address prefixes. Prior work has expanded addresses into prefixes indiscriminately [58, 77, 82] – this improves recall but greatly decreases specificity, as we show in Section 6.1. The novelty of our approach is in first identifying dynamic and mismanaged networks and then expanding addresses belonging to these networks into prefixes, only when this expansion is likely to bring us higher gain in recall than loss in specificity.

The expansion phase starts with master blacklist produced in the previous step, and calculates which addresses could be expanded into their /24 prefixes (see Section 6 for rationale behind choosing /24 prefix size). We first generate a list of all /24 prefixes that contain addresses on the master blacklist. We then evaluate if each prefix is either dynamically addressed or mismanaged. On a positive finding, we estimate the recall gain and specificity loss, using our historical offense and safety scores as proxies, respectively. If the estimated gain exceeds the loss, we will include the /24 prefix on the master blacklist.

4.3.1 *Mismanaged Prefixes*

We classify a /24 prefix as potentially mismanaged if it contains more than the threshold N_m addresses, reported by any blacklist. We set $N_m = 2$ (see Section **??** for rationale).

Table 3 shows the number of addresses, which are either dynamically addressed or mismanaged or both. About 22.4% of networks are only dynamic networks, 10.6% are both dynamic and mismanaged, 42.4% are only static networks and 24.6% are static and mismanaged. Overall, 57.6% of addresses on our master blacklist are candidates for expansion into prefixes.

4.3.2 *Selective Expansion*

If a /24 prefix is found to be either a dynamic or mismanaged, we selectively expand such addresses into their corresponding /24 prefixes. We estimate the cost of expansion for the /24 prefix a, denoted by SE_a, as:

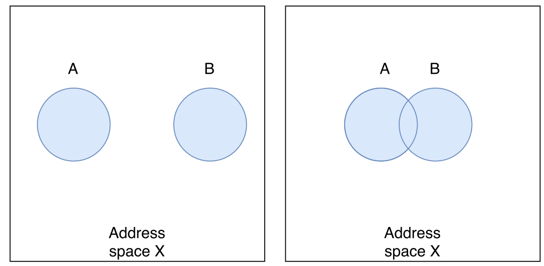
$$SE_a = 1 - \frac{F_a}{P_a + F_a} \quad (6)$$

where P_a is the number of addresses from address space a, which are predicted to be malicious (i.e., which are included in the BLAG's master blacklist) and F_a is the number of legitimate addresses, which have been historically misclassified as malicious, from Equation 3.

We define a threshold β such that, addresses a which are either dynamic or mismanaged are expanded if $SE_a \geq \beta$. We discuss the choice of β in Section **??**.

Type	**# of Networks**	**% of Networks**
Only dynamic	312,169	22.4%
Dynamic and mismanaged	148,584	10.6%
Only static	592,829	42.4%
Static and mismanaged	343,370	24.6%
Total	1,396,952	100%

Table 3: Breakdown of probed networks into static, dynamic or mismanaged.



(a) Case 1 (b) Case 2

Figure 5: Different use cases, illustrating how and why BLAG works.

4.4 Why and How BLAG Works

BLAG assigns reputation scores to addresses listed in blacklists. An address can have a low reputation score when the blacklist listing the address is not safe enough for that address space ($s_{a,b}$ is low) or when the address may not have been recently listed in the blacklist ($ho_{a,b}$ is low) or when both $s_{a,b}$ and $ho_{a,b}$ are low. In such cases, these addresses will have a smaller chance to propagate to the aggregated master blacklist. This may be the right decision in some cases, while in others sparsity of the reputation matrix may impair timely inclusion of repeat offenders on the master blacklist. Recommendation system helps overcome the sparsity problem. We now illustrate with a few toy examples how BLAG works, and how it can achieve smart aggregation. Figure 5 illustrates these cases. For simplicity, imagine that there are only two blacklists reporting address sets A and B in some address region X. **Case 1: Aggregating disjoint**

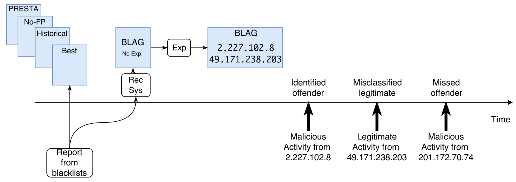


Figure 6: Determining when an address (offender or legitimate) is identified correctly or incorrectly by a blacklist.

listings. Our reputation scores help identify accurate (timely and safe) listings in A and B. When A and B are disjoint, the recommender system cannot help fill in empty cells. Only addresses that already have a high score in A and B will be included in the master blacklist.

Case 2: Aggregating partially disjoint listings. Our reputation scores help identify accurate listings in A and B. When A and B have some overlap, the recommender system helps fill in empty cells (i.e., those in A but not in B and those in B but not in A). Addresses that already had a high score in A or B will be included in the master blacklist – the recommender system does not influence their inclusion. Other addresses that received a high score by the recommender system (in empty cells of score matrix) but originally had a low score in A or B benefit from the recommender system. They will be included in the aggregated master blacklist, whereas they would have been left out otherwise. In a sparse matrix, many addresses may be influenced this way, and promoted for inclusion. Finally, addresses that received a low score by the recommender system, and that originally had a low score, will be excluded. Recommender system does not influence their exclusion.

As BLAG assimilates more blacklists, the probability of an address matching the case 2 instead of case 1 grows, and we reap benefits from the recommender systems. We evaluate this effect in Section 5. Unfortunately, an attacker could misuse the recommender system to pollute our master blacklist. We discuss this effect and possible solutions in Section 8.

5. EVALUATION

In this Section we evaluate the performance of BLAG and several competing approaches, using datasets described in Section 2. We find that BLAG achieves reasonably high specificity (95–97%) and high recall (11–65%), while other approaches have either very low recall (<1%), or higher recall but low specificity (≈84%).

5.1 Evaluation setup

Competing approaches: We compare BLAG's performance against four approaches:

1. *Best* – the blacklist from Black dataset that performs the best on a given ground-truth dataset with regard to recall; we start with the most recent snapshot of that blacklist prior to the start of the ground-truth dataset and then refresh it with snapshots taken during the ground-truth dataset. *Best* is a hypothetical scenario, because in real deployment we could not tell which blacklist will eventually be the best. It allows us to measure benefits of aggregation over use of a single blacklist.

2. *Historical* – all addresses listed in any blacklist in the Black dataset, up until the end of a given ground-truth dataset. This approach assumes "once malicious, always malicious".

3. *PRESTA* – the blacklist generated using technique described in [83]. PRESTA performs spatio-temporal analysis and expansion using historical blacklist data to generate a more proactive blacklist. It expands some addresses, which are repeat or recent offenders, into their /24 prefix, and two surrounding /24 prefixes. For example, if the address 1.2.3.4 were chosen for expansion, PRESTA would include 1.2.2.0/24, 1.2.3.0/24 and 1.2.4.0/24 in its blacklist. [Sivaram: NDSS reviewers had issue with this approach and said that it wasnt a fair evaluation. Can we just use PRESTA's expansion technique for comparison?]

Figure 6 illustrates our evaluation methodology. We first take each of our three ground-truth datasets and divide it into seven days of training and the rest is used for testing for email and scanning dataset. For DNS dataset, we use one day of training and one day for testing. We use the legitimate part of the training dataset as our (L) set. During evaluation, for our testing set and for each blacklisting approach (best, historical or BLAG) we simulate the dynamics with which the addresses appear over time, both in individual blacklists (Black) and in ground-truth datasets. When an address appears in the Black dataset we:

- include the address in the best blacklist if it appeared on a blacklist, which will ultimately perform the best on the given malicious dataset,
- include the address in the historical blacklist,
- apply PRESTA algorithm on the address to determine if it is included in the PRESTA blacklist,
- apply BLAG on the address to determine if it should be included in the BLAG's aggregated master blacklist, and if it should be expanded into its /24 prefix.

Evaluation metrics: In our evaluation we measure recall and specificity for each approach as follows. (1) *Specificity:* When a legitimate address l appears in our testing dataset, we compare its address against the currently generated blacklist (competing approach or BLAG) at the same point in time in our simulation. If listed on that blacklist, we count l as a false positive. Specificity is the percentage of legitimate addresses that were not false positives. (2) *Recall:* When an offender o commits offense in our testing dataset, we compare the offender's address against the currently generated

Parameter	Description	Value
l	Length of address history	30 days
K	Number of latent features	5
α	Reputation threshold	8

Table 4: Parameters used in evaluation.

blacklist (competing approach or BLAG) at the same point in time in our simulation. We mark o as true positive if it is currently listed by the blacklist. Recall is the percentage of offenders that are true positives.

Parameter settings: Parameters used in our evaluation are summarized Table 4. We set length of address history $l = 30$, number of latent features for recommendation system $K = 5$ and reputation threshold $\alpha = 8$. Our choices for these variables are explained in Section 6.

5.2 BLAG is More Accurate

The goal of reactive blacklisting is to capture as many offenders as possible, while keeping the specificity high. [Sivaram: Do you think the representation of the figs is better? The earlier format did not show much difference in specificity, as the values of specificities were usually high and recall low.]

BLAG has the best specificity and recall across three traffic datasets: Figure 7 shows that BLAG has overall the best performance. Best blacklist has the highest specificity (99.9–100%) but the lowest recall (0.02–0.1%), indicating that a single blacklist is lot enough to capture all attackers. Historical blacklist, which is naive aggregation of all blacklist data has better recall (0.7–18.3%) but has lower specificity (88.8–93.7%) than best blacklist. BLAG's specificity is uniformly high (94.9–97.4%) and its recall (11.6–65.6%) is higher than that of best and historical blacklists.

BLAG's performance comes both from selection of high-quality data for aggregation and from expansion of addresses into prefixes. We investigated how much of BLAG's performance comes from its selection of high-quality data to aggregate and how much comes from expansion, by showing BLAG with and without expansion (*BLAG* and *BLAG No Exp* in Figure 7). We compare this performance to performance of best blacklist, and to historical which perform naive aggregation without expansion.

Even without expansion, BLAG achieves recall(0.7–17.8%), which is always better than best, with a small loss of specificity (1–2.4%). Historical blacklists have slightly better recall (0.6–5.2%) than BLAG without expansion, but lose up to 6.2–11.1% of specificity. Thus, smart aggregation helps BLAG improve recall and specificity over naive and no aggregation approaches. Expansion of BLAG then improves recall further, at a small loss of specificity (up to 5% loss on our traffic datasets). We show in Section 6 that, even if we applied selective expansion on other blacklisting approaches, BLAG would still outperform them.

BLAG filters more attacks. Some addresses may generate more attacks than others, i.e., they could be more malicious. We evaluate the amount of malicious activity (e.g., spam, scanning, etc) that would be filtered by BLAG, best and historical blacklists for our three traffic datasets. In case of email dataset, BLAG would filter 65.6% of spam, compared to 0.19% and 18.3% filtered by best and historical blacklists respectively. In case of scanning dataset, BLAG would filter traffic from 56.3% of infected devices, compared to only 0.07% and 9.4% filtered by best and historical blacklists. In case of DNS dataset, BLAG would drop 11.6% of attackers, compared to 0.0002% and 0.7% filtered by best and historical blacklists respectively.

6. SENSITIVITY ANALYSIS

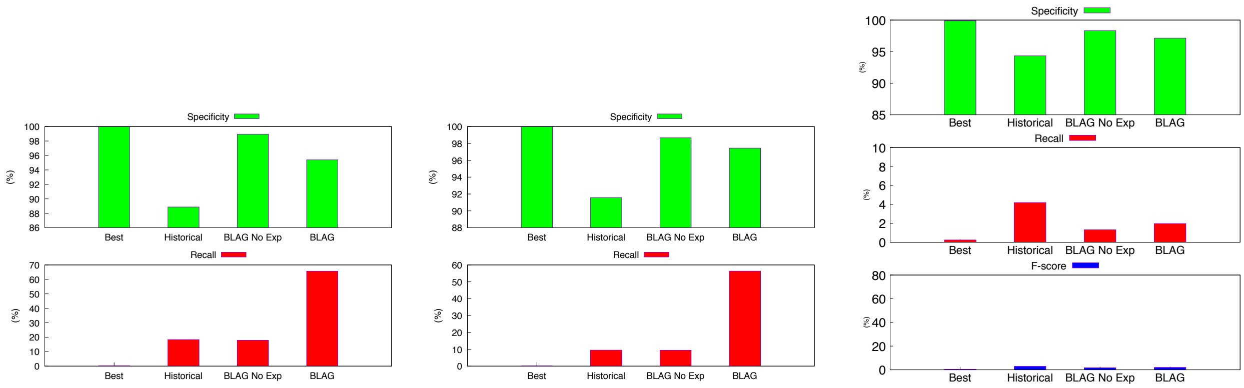
In this section we discuss our design choices and values we adopt for constants l, K and α.

6.1 Expansion Approach

BLAG expands select addresses into /24 prefixes. In this subsection we investigate two questions. First, we ask what if a similar expansion approach to BLAG's were applied to best blacklist and historical blacklist. Instead of selective expansion here, which uses BLAG's information, we use regular expansion, where each candidate address is expanded into its /24 prefix only when there are no other addresses present in the training dataset. We show that BLAG still outperforms competing approaches, due to its selection of only high-quality information to aggregate, prior to expansion. Finally, we investigate how BLAG's performance would change if we expanded addresses into their full BGP prefixes or entire autonomous systems.

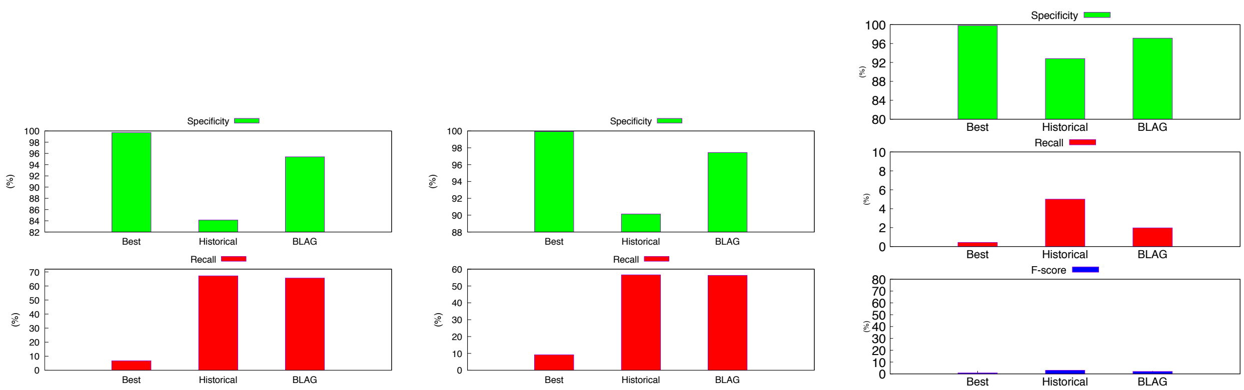
BLAG outperforms best-expanded and Historical-expanded blacklists. We compare BLAG to best and historical blacklists, with regular expansion and show their performance in Figure 8. Historical expanded blacklists have 0.5%–13.6% better recall than BLAG for all the three datasets, but at the loss of up to 15.8% of specificity. BLAG has higher recall (11.6–65.6%) than the best-expanded blacklists, which has only 0.1–9.1% recall. BLAG achieves this at modest specificity cost. BLAG loses up to 2.5–5% of specificity, while best-expanded loses 0.1% and historical-expanded loses 9.8–15.8%. Thus BLAG strikes the right balance between maintaining high specificity and improving recall.

BLAG's expansion approach outperforms expansion at BGP prefix and AS level. Previous work suggested aggregating addresses into BGP prefixes [77]. We apply /24 prefix, BGP prefix and AS level aggregation to the master blacklist, produced by BLAG for the the three datasets. We apply the selective expansion technique, but instead of expanding to /24 prefix, we expand to BGP prefix and AS. We show their recall and specificity in Figure 9. [Sivaram: By using the selective expansion technique we do get better results in capturing attackers. However, I still believe that blocking entire BGP prefix/ASN can be controversial and can raise concerns among reviewers. Do you think, I should



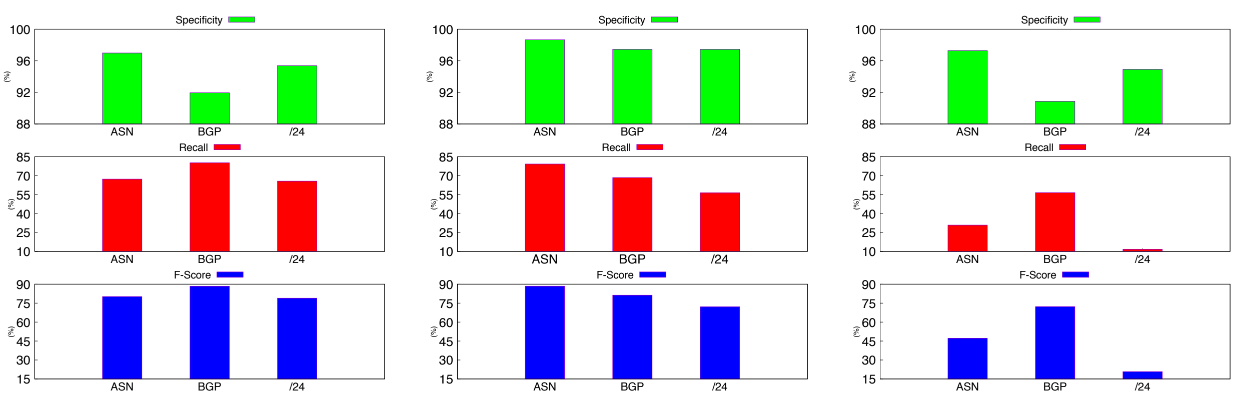
(a) Email (b) Scanning (c) DNS

Figure 7: Specificity and recall of BLAG with two competing approaches on traffic datasets.



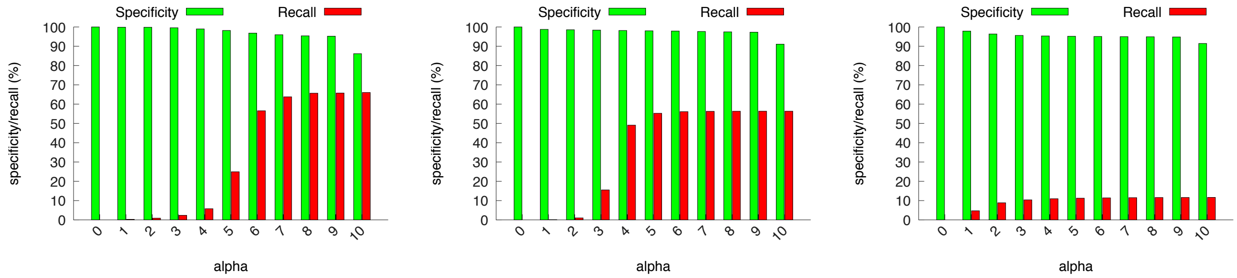
(a) Email (b) Scanning (c) DNS

Figure 8: Specificity and recall of BLAG and four competing approaches with expansion.



(a) Email (b) Scanning (c) DNS

Figure 9: Evaluating BGP and AS expansion techniques.



(a) Email (b) Scanning (c) DNS

Figure 10: Evaluating α

just expand blindly into BGP prefix/ASN without the selective expansion?]

6.2 Contribution of Individual Blacklists by Size

We ran BLAG on n largest blacklists, and varied n from 1 to 157 . We report the specificity and recall of these tests on email dataset in Figure 11. We observe that the recall increases as we add more blacklists but it has diminishing returns. There is 15.6% gain in recall for the first 106 blacklists. After which, there is a sharp increase in recall for the remaining blacklists. The top 10 largest blacklists have recall of 13.2%, top 40 have have a recall of 40.1%, top 80 have a recall of 62% and all blacklists have a recall of 65.6%. As we add more blacklists, specificity drops slightly from 100% with first 100 blacklist, to 98% when top 100 blacklists are used, drops further to 97.1% for the next 30 blacklists and finally to 95.3% when all blacklists are used. This illustrates that having a moderate number (several tens) of blacklists would likely suffice to reap benefits of running BLAG.

6.3 Parameter l for Historical Decay

[Sivaram: I am thinking of showing for ten different values of l from 10 to 110 days for the three datasets. What do you think?]

6.4 Parameter K for Matrix Factorization

[Sivaram: I think I am hand waving here. Though it K is relevant in matrix factorisation, I make sure that the RMSE drops below 1% for the iterations to stop. What do you think would be a good evaluation for this? Do you think we need one?] A critical parameter in non-negative matrix factorization (NMF) used in BLAG is the parameter K, which is the number of latent features. An ideal K will have the minimum error between the matrix $R(MxN)$ and the cross product of P and Q (see Section 4.2). Brunet et al. [54] suggested in using the smallest K, after which the *cophenetic correlation coefficient* starts decreasing. We evaluate different values of K by considering the most dense /16 prefix, consisting of 56,102 addresses, which are present in over 67 blacklists. We vary K from 2 to 67 ($K \leq min(M, N)$) and find that cophenetic correlation coefficient starts decreasing after K is 3.

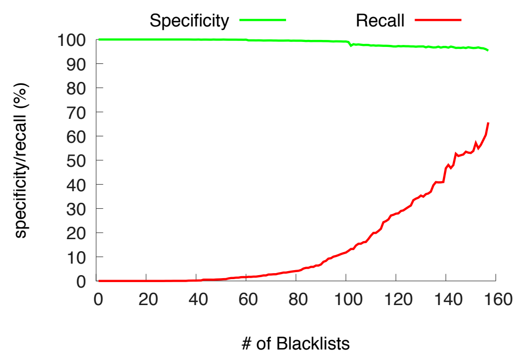


Figure 11: Varying size of blacklists for mailinator dataset.

We ran gradient descent with $K = 3$ until the root mean squared error (RMSE) between the original matrix R and matrix R' (obtained after the cross product of P and Q) fell below 1% or the number of iterations exceeded 20,000. We observed that 98.7%, 99.37% and 95.9% of prefixes in Mailxam, Miraixa and Darkexa datasets have RMSE less than 1%.

6.5 Parameter α for Choosing Addresses

Parameter α controls the set of addresses, which should be considered for the expansion phase in BLAG. Figure 10 shows that for each dataset parameter α trades in accuracy of BLAG with higher coverage. We see that as value of α increases, BLAG's recall increases but specificity drops. We set α to 8 for all the three datasets. We observe that recall changes from 0.2% to 65.6%, 0.1% to 56.3% and 4.7% to 11.6% for email, scanning and DNS dataset respectively. While the specificity increases by 1.3–4.6% across the three datasets. Higher values of α (9–10), bring less improvement in recall with much higher loss in specificity. We observe that recall increases only by 0.1–0.5%, while specificity reduces by 3.7–10.7% for all the three datasets.

7. RELATED WORK

In this Section we survey work related to blacklist analysis, creation or aggregation.

Analysis of Blacklists. Kührer et al. evaluated the effectiveness of fifteen publicly available malware blacklists by measuring accuracy and completeness on datasets consisting of parked domains, sinkholed addresses and active malware domains [62]. Pitsillidis et al. evaluated ten different blacklists on purity, coverage, proportionality and timing [70]. Purity was measured by comparing feeds to Alexa top websites and Open directory listing, whereas coverage, proportionality and timing were obtained by comparing feeds to one another. Both Kührer et al. and Pitsillidis et al. works support our own findings that blacklists are not accurate. Zhang et al. evaluated nine blacklists using traffic logs from a regional ISP [86]. They analyzed overlapping addresses between traffic logs and blacklists. But they were unable to measure the accuracy of blacklists, as the traffic in the logs was not labeled as malicious or legitimate. The main difference between these related works and ours is twofold. First, our main focus is on distilling accurate information from blacklists and aggregating it into a master blacklist; we use data about current blacklists performance merely to motivate our work. Second, we use an order of magnitude more blacklists than previous works.

Improving Blacklisting. Highly Predictive Blacklisting [85] (HPB) proposes a technique to create blacklists customized to the given network. The algorithm is based on an algorithm similar to Google's page ranking scheme, which identifies attackers that may target the specific customer, based on the attacks reported by other similar customers. Though this produces effective blacklists for a given network, HPB will not uncover new attackers, which have not targeted a specific customer group, while BLAG does not have this limitation. Soldo et al. [78] built on HPB. They extended it to use historical data about attack sources and destinations, and to use a recommendation system to predict possible attackers given a victim. In contrast, BLAG does not produce a victim-specific blacklist, but a generic one. Our recommendation system does not learn affinity between attackers and victims, but between attackers and blacklists, and we use it to improve recall, while keeping the specificity high. We expect that BLAG's blacklists may be able to filter more attacks than Soldo et al. approach.

There are several works which focus on improving spam mitigation using new blacklisting techniques. PRESTA [83] uses historical data from three spam blacklists provided by Spamhaus [44], to infer temporal and spatial properties of addresses and expand addresses into spatial regions, similar to our /24 prefixes. PRESTA does not consider the possibility of false positives from the expansion, which leads to lower specificity compared to BLAG, as seen in Section 5. Sinha et al. [77] present how to improve spam blacklisting, by using a thresholding technique, which includes the number of sent messages into spammer identification process. They also present an expansion technique, which blacklists the entire BGP prefix if that prefix sent only spam. Though this technique is effective in detecting new spammers, expansion to BGP prefixes is too coarse-grained and can lower specificity, as shown in Section 6.

We could not directly compare Soldo et al. and Sinha et al. approaches to BLAG, because both these approach need data on attackers and victims, which is not publicly available.

8. DISCUSSION

In this section we discuss possible attacks on BLAG, and some deployment issues.

Pollution. BLAG has no way, other than the reputation system, to differentiate between low-quality and high-quality information. Thus, if an attacker could produce a blacklist that is very similar to some reputable blacklist (e.g., by copying it) and if he included a few public servers in it, BLAG could conceivably propagate this information into its master blacklist. This could then lead to legitimate traffic from these public servers being dropped. Current blacklists could also be polluted by the same approach. BLAG makes polluted information less likely to propagate, than the use of individual blacklists. The attacker would have to carefully craft the polluted blacklist so that the servers reside in the same /16 as many malicious hosts; otherwise BLAG would be able to identify and discard low-quality information. Similarly, the attacker would have to insert just one or a few servers into these /16 ranges, or else their safety score would be too low for inclusion in the master blacklist. This means that the attacker cannot manipulate BLAG's master blacklist at will, but can just target those legitimate clients who share a /16 range with many recently malicious hosts. While limited, this effect is still very undesirable and we would like to prevent it.

BLAG can monitor the quality of each blacklist, e.g., how many misclassifications each blacklist usually makes on the second known-legitimate dataset $L\prime$. If a sudden increase is detected, BLAG can use machine unlearning [55] to selectively remove this list's historical data from the final blacklist. Since BLAG processes data for individual /16 networks, only the reputation scores for the affected /16 networks would need to be recomputed. We leave the exact handling of pollution attempts for our future work.

Frequency of running BLAG. Initially, BLAG would start with some set of blacklists and it would be ran over their current and historical data. Once the reputation scores are computed, BLAG needs to be run only when a blacklist is updated. When addresses are added or removed in a snapshot, only the reputation scores for the encompassing prefixes must be recalculated.

Overhead of running BLAG. Blacklists could be updated very often, which may lead to large overhead to generate master blacklist. In our evaluation, there were about 4,000 updates per day and they took around 3 hours to process on a 4-core, 16GB RAM server. Since, BLAG evaluates blacklists at a level of /16 prefix, BLAG operation can be distributed among several machines to achieve scalability.

How can BLAG be used. Current blacklists are used to pre-filter known offender traffic, i.e., they are used proactively. BLAG's master blacklist can be used in the same manner. But because BLAG's specificity is lower than that of individual blacklists, one could also decide to use it reactively, to prioritize which traffic should be dropped. For example, when there is a heavy DDoS attack, when a worm is spreading, or when a new vulnerability becomes known. BLAG can also be used with ensemble filtering systems such as SpamAssassin [65] where multiple detection techniques are used in parallel on an email to decide if it is a spam or ham. In such cases, BLAG can become one of the detection techniques.

9. CONCLUSION

Blacklists are widely used by network operators, but they usually miss many attacks. We have proposed BLAG– the system that can identify high-quality pieces of information from multiple blacklists, and aggregate them into a master blacklist, with some addresses expanded into /24 prefixes. Overall, BLAG has a higher recall than any single blacklist or their naive aggregation with minimal loss in specificity. BLAG also outperforms PRESTA, a competing approach, by having much higher specificity. We thus believe that BLAG could significantly improve network security, and lower collateral damage to legitimate traffic from blacklisting.

10. ACKNOWLEDGEMENT

This material is based on research sponsored by the Department of Homeland Security (DJ-IS) Science and Technology Directorate, Homeland Security Advanced Research Projects Agency (HSARPA), Cyber Security Division (DHS S&T/HSARPACSD) BAAHSI-IQDC-14-R-B0005, and the Government of United Kingdom of Great Britain and Northern Ireland via contract number DI 5PCOO 184. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Department of Homeland Security, the U.S. Government, or the Government of United Kingdom of Great Britain and Northern Ireland.

11. REFERENCES

[1] Alienvault Reputation System. https://www.alienvault.com/.
[2] Asprox tracker. http://atrack.h3x.eu/.
[3] Automated ip reputation system for spammers. http://www.chaosreigns.com/iprep/.
[4] badips.com — an ip based abuse tracker. https://www.badips.com/.
[5] Bambenek consulting feeds. http://osint.bambenekconsulting.com/feeds/.
[6] Binary defense systems — defend. protect. secure. https://www.binarydefense.com/.
[7] Blocklist.de fail2ban reporting service. https://www.blocklist.de/en/index.html.
[8] Charles b. haley. http://charles.the-haleys.org/.
[9] Cinsscore.com. http://cinsscore.com/.
[10] Cisco talos - additional resources. http://www.talosintelligence.com/additional-resources/.
[11] Clean mx - realtime db. http://www.clean-mx.com.
[12] Cloud spam protection for forums, boards, blogs and sites. https://www.cleantalk.org.
[13] Cloudflare inc. https://www.cloudflare.com.
[14] Cyber threat alliance. http://cyberthreatalliance.org/.
[15] Cybercrime. http://cybercrime-tracker.net/.
[16] Daniel gerzo bruteforceblocker. http://danger.rulez.sk/index.php/bruteforceblocker/.
[17] Darklist.de — a malicious ip catcher blacklist. http://darklist.de/.
[18] Emerging Threats. https://rules.emergingthreats.net/.
[19] Emerging threats. https://rules.emergingthreats.net/fwrules/emerging-Block-IPs.txt.
[20] Greylist :: Project:turris. https://www.turris.cz/en/greylist.
[21] hphosts - by malware bytes. https://hosts-file.net/.
[22] I-blocklist — home. https://www.iblocklist.com/.
[23] Index of /pub/malware-feeds/. http://security-research.dyndns.org/pub/malware-feeds/.
[24] Ip blacklist — graphiclineweb. https://graphiclineweb.wordpress.com/tech-notes/ip-blacklist/.
[25] jigsaw-security — open blacklist. http://www.jigsawsecurityenterprise.com/open-blacklist.
[26] Keyword research, competitor analysis, & website ranking — alexa. http://www.alexa.com/.
[27] Krebsonsecurity hit with record ddos. https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/.
[28] Lashback blacklist. http://blacklist.lashback.com/.
[29] Mailinator inc. http://www.mailinator.com.
[30] Malc0de Database. http://malc0de.com/database/.
[31] Malware domain list. http://www.malwaredomainlist.com/.
[32] Mirai: The iot bot that took down krebs and launched a tbps ddos attack on ovh.

https://goo.gl/GH3Nxc.
[33] My ip - blacklist checks. https://www.myip.ms/info/about.
[34] Netlab - a quick stats on the 608,083 mirai ips that hit our honeypots in the past 2.5 months. https://goo.gl/NYWMLq.
[35] Nothink individual blacklist maintainer. http://www.nothink.org/.
[36] Novirusthanks: Security software and services. http://www.novirusthanks.org/.
[37] Openbl.org - abuse reporting and blacklisting. https://www.openbl.org/.
[38] Project Honeypot. https://www.projecthoneypot.org/.
[39] Pushing inertia by dan traczynski - online presence of a vancouver, canada software professional. http://pushinginertia.com/.
[40] Sample list of high risk ip addresses — maxmind. https://www.maxmind.com/en/high-risk-ip-sample-list.
[41] Sblam! zabezpiecza formularze przed spamem. http://sblam.com/.
[42] Server blocklist / blacklist - cruzit.com - php, linux & dns tools, apache, mysql, postfix, web & email spam prevention information. http://www.cruzit.com/wbl.php.
[43] Sourcefire vrt labs. https://labs.snort.org/.
[44] Spamhaus Project. https://www.spamhaus.org/.
[45] The spamhaus project - frequently asked questions (faq). https://www.spamhaus.org/faq/section/Spamhaus%20SBL#7.
[46] Stefan gofferje - sip attacker blacklist. http://stefan.gofferje.net/it-stuff/sipfraud/sip-attacker-blacklist.
[47] Stop forum spam. https://stopforumspam.com/.
[48] Swiss Security Blog - Abuse.ch. https://www.abuse.ch/.
[49] Trustedsec - information security consulting services. https://www.trustedsec.com/.
[50] Voip blacklist. http://www.voipbl.org/.
[51] We catch bots so that you don't have to. https://www.botscout.com.
[52] - blocklist.net.ua. https://blocklist.net.ua/.
[53] Ross Anderson, Chris Barton, Rainer Böhme, Richard Clayton, Michel JG Van Eeten, Michael Levi, Tyler Moore, and Stefan Savage. Measuring the cost of cybercrime. In *The economics of information security and privacy*, pages 265–300. Springer, 2013.
[54] Jean-Philippe Brunet, Pablo Tamayo, Todd R Golub, and Jill P Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the national academy of sciences*, 101(12):4164–4169, 2004.
[55] Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 463–480. IEEE, 2015.
[56] Martin Casado and Michael J Freedman. Peering through the shroud: The effect of edge opacity on ip-based client identification. In *Proceedings of the 4th USENIX conference on Networked systems design & implementation*, pages 13–13. USENIX Association, 2007.
[57] Alberto Dainotti, Alistair King, and Kimberly Claffy. Analysis of internet-wide probing using darknets. In *Proceedings of the 2012 ACM Workshop on Building analysis datasets and gathering experience returns for security*, pages 13–14. ACM, 2012.
[58] Shuang Hao, Nadeem Ahmed Syed, Nick Feamster, Alexander G Gray, and Sven Krasser. Detecting spammers with snare: Spatio-temporal network-level automatic reputation engine. In *USENIX security symposium*, volume 9, 2009.
[59] SANS Institute. Internet Storm Center. https://dshield.org/about.html.
[60] Justin K Keane. Using the google safe browsing api from php. *Mad Irish, Aug*, 7, 2009.
[61] Yehuda Koren, Robert Bell, Chris Volinsky, et al. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.
[62] Marc Kührer, Christian Rossow, and Thorsten Holz. Paint it black: Evaluating the effectiveness of malware blacklists. In *International Workshop on Recent Advances in Intrusion Detection*, pages 1–21. Springer, 2014.
[63] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.
[64] Gregor Maier, Anja Feldmann, Vern Paxson, and Mark Allman. On dominant characteristics of residential broadband internet traffic. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 90–102. ACM, 2009.
[65] Justin Mason. Filtering spam with spamassassin. In *HEANet Annual Conference*, page 103, 2002.
[66] Heise Online. Nixspam Blacklist. https://goo.gl/jsyksA.
[67] Ramakrishna Padmanabhan, Amogh Dhamdhere, Emile Aben, Neil Spring, et al. Reasons dynamic addresses change. In *Proceedings of the 2016 ACM on Internet Measurement Conference*, pages 183–198. ACM, 2016.
[68] Vern Paxson. Bro: a system for detecting network intruders in real-time. *Computer networks*, 31(23):2435–2463, 1999.
[69] Michael Pazzani and Daniel Billsus. Content-based

recommendation systems. *The adaptive web*, pages 325–341, 2007.

[70] Andreas Pitsillidis, Chris Kanich, Geoffrey M Voelker, Kirill Levchenko, and Stefan Savage. Taster's choice: a comparative analysis of spam feeds. In *Proceedings of the 2012 ACM conference on Internet measurement conference*, pages 427–440. ACM, 2012.

[71] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. In *ACM SIGCOMM Computer Communication Review*. ACM.

[72] Anirudh Ramachandran, Nick Feamster, and Santosh Vempala. Filtering spam with behavioral blacklisting. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 342–351. ACM, 2007.

[73] Philipp Richter, Georgios Smaragdakis, David Plonka, and Arthur Berger. Beyond counting: New perspectives on the active ipv4 address space. *arXiv preprint arXiv:1606.00360*, 2016.

[74] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.

[75] Dominik Schatzmann, Martin Burkhart, and Thrasyvoulos Spyropoulos. Inferring spammers in the network core. In *International Conference on Passive and Active Network Measurement*, pages 229–238. Springer, 2009.

[76] Sushant Sinha, Michael Bailey, and Farnam Jahanian. Shades of grey: On the effectiveness of reputation-based blacklists. In *Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on*, pages 57–64. IEEE, 2008.

[77] Sushant Sinha, Michael Bailey, and Farnam Jahanian. Improving spam blacklisting through dynamic thresholding and speculative aggregation. In *NDSS*, 2010.

[78] Fabio Soldo, Anh Le, and Athina Markopoulou. Predictive blacklisting as an implicit recommendation system. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. IEEE, 2010.

[79] Brett Stone-Gross, Marco Cova, Lorenzo Cavallaro, Bob Gilbert, Martin Szydlowski, Richard Kemmerer, Christopher Kruegel, and Giovanni Vigna. Your botnet is my botnet: analysis of a botnet takeover. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 635–647. ACM, 2009.

[80] Kurt Thomas, Rony Amira, Adi Ben-Yoash, Ori Folger, Amir Hardon, Ari Berger, Elie Bursztein, and Michael Bailey. The abuse sharing economy: Understanding the limits of threat exchanges. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 143–164. Springer, 2016.

[81] Kurt Thomas, Danny Yuxing, Huang David, Wang Elie, Bursztein Chris Grier, Thomas J Holt, Christopher Kruegel, Damon McCoy, Stefan Savage, and Giovanni Vigna. Framing dependencies introduced by underground commoditization. In *In Proceedings (online) of the Workshop on Economics of Information Security (WEIS*. Citeseer, 2015.

[82] Shobha Venkataraman, Subhabrata Sen, Oliver Spatscheck, Patrick Haffner, and Dawn Song. Exploiting network structure for proactive spam mitigation. In *Usenix Security*, 2007.

[83] Andrew G West, Adam J Aviv, Jian Chang, and Insup Lee. Spam mitigation using spatio-temporal reputations from blacklist history. In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 161–170. ACM, 2010.

[84] Nicky Woolf. DDoS attack that disrupted internet was largest of its kind in history, experts say. `https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet`, 2016.

[85] Jian Zhang, Phillip A Porras, and Johannes Ullrich. Highly predictive blacklisting. In *USENIX Security Symposium*, pages 107–122, 2008.

[86] Jing Zhang, Ari Chivukula, Michael Bailey, Manish Karir, and Mingyan Liu. Characterization of blacklists and tainted network traffic. In *International Conference on Passive and Active Network Measurement*, pages 218–228. Springer, 2013.

[87] Jing Zhang, Zakir Durumeric, Michael Bailey, Mingyan Liu, and Manish Karir. On the mismanagement and maliciousness of networks. In *NDSS*, 2014.