BLAG: Improving the Accuracy of Blacklists

Sivaramakrishnan Ramanathan
University of Southern California/
Information Sciences Institute
satyaman@usc.edu

Jelena Mirkovic
University of Southern California/
Information Sciences Institute
mirkovic@isi.edu

Minlan Yu
Harvard University
minlanyu@g.harvard.edu

***Abstract*—IP address blacklists are a useful source of information about repeat attackers. Such information can be used to prioritize which traffic to divert for deeper inspection (e.g., repeat offender traffic), or which traffic to serve first (e.g., traffic from sources that are not blacklisted). But blacklists also suffer from overspecialization – each list is geared towards a specific purpose – and they may be inaccurate due to misclassification or stale information. We propose BLAG, a system that evaluates and aggregates multiple blacklists feeds, producing a more useful, accurate and timely *master blacklist*, tailored to the specific customer network. BLAG uses a sample of the legitimate sources of the customer network's inbound traffic to evaluate the accuracy of each blacklist over regions of address space. It then leverages recommendation systems to select the most accurate information to aggregate into its master blacklist. Finally, BLAG identifies portions of the master blacklist that can be expanded into larger address regions (e.g. /24 prefixes) to uncover more malicious addresses with minimum collateral damage. Our evaluation of 157 blacklists of various attack types and three ground-truth datasets shows that BLAG achieves high specificity up to 99%, improves recall by up to 114 times compared to competing approaches, and detects attacks up to 13.7 days faster, which makes it a promising approach for blacklist generation.**

I. Introduction

IP blacklists ("blacklists" for short), which contain identities of prior known offenders, are usually used to aid more sophisticated defenses, such as spam filters or security information and event management (SIEM) systems, in identifying traffic that warrants further analysis [15]. Blacklists do not have to be very accurate to be used in this *advisory* mode, since they merely aid another, more sophisticated system, which decides traffic's destiny. In fact, prior works [66], [60], [63], [70], [74] show that blacklists are often not very accurate or useful. They do not misclassify many legitimate sources, but they miss the majority of malicious sources.

What if a network suffers a novel attack, which bypasses its sophisticated defenses? Or what if the attack has such a large volume that the more sophisticated system cannot keep up? In these situations, networks usually resort to crude defenses, such as rate-limiting traffic or filtering all incoming traffic to a given port number or a destination [65], [67], [1], [54].

IP blacklists could be used *as emergency response*, in case of a novel or large-scale attacks, to filter attacks from prior known malicious sources, and act as the first layer of defense. For example, an email server hit by a heavy phishing campaign, whose signature does not yet exist in the server's spam filters, could use blacklists to drop emails sent by prior known malicious sources. Or a network under a distributed denial-of-service attack could use blacklists to drop all traffic sent by prior known malicious sources, to lessen its load. Blacklists are easy to implement since only the source IP address of incoming traffic is checked, which can be cheap at line speed. In this emergency mode, blacklists would have to be very accurate, since they would actively drop traffic. This means that blacklists should be able to identify the *majority of attack sources* while keeping *misclassification of legitimate sources low*. This paper explores how to generate sufficiently accurate blacklists for emergency use.

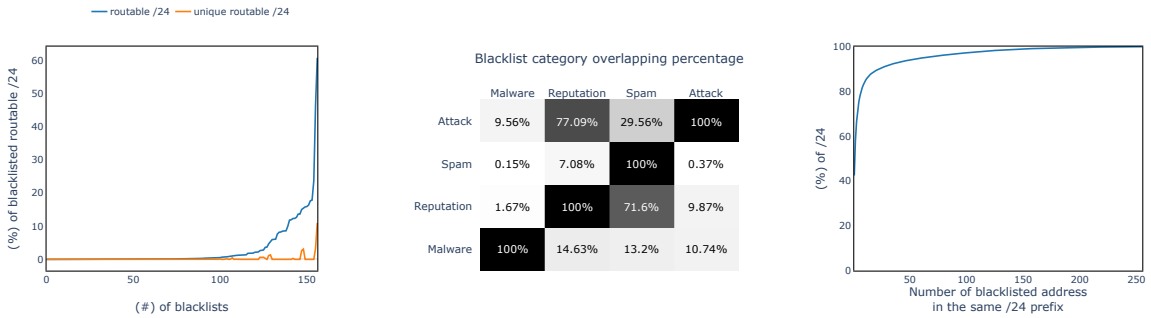
Individual blacklists today suffer from several drawbacks that limit their accuracy in malicious source identification. Firstly, individual blacklists miss many malicious sources. This effect may come from their limited vantage points – e.g., blacklist maintainers may have honeypots in the United States but not in India – or from their limited scope – e.g., blacklists are created for specific attack classes, like spam. On the other hand, compromised devices are constantly being drafted into botnets and misused for different attacks, such as sending spam one day and participating in denial-of-service attacks on a different day. *Aggregating blacklists from different maintainers and across various attack types* can improve the accuracy of malicious source identification over any individual blacklist.

Secondly, blacklists are snapshots of malicious sources at a given time. Attackers are known to evade blacklisting by staying dormant, only to resume malicious activities later [66]. *Historical blacklist data can provide additional intelligence* on active past offenders that are likely to re-offend in the future.

Finally, malicious sources have historically been known to concentrate in a few mismanaged networks [81]. Thus, *expanding certain blacklisted IP addresses into IP prefixes* could improve the accuracy of malicious source identification. But the aggregation of data from multiple lists and past periods, and expansion of addresses into prefixes may greatly increase misclassifications of legitimate traffic *if applied naïvely*. We propose BLAG (**BL**acklist **AG**gregator), which performs *smart aggregation of blacklist data*, and tailors this data to the customer network[1]. BLAG's blacklists have a much higher accuracy of malicious source identification and they keep collateral damage to legitimate sources low. BLAG overcomes problems of existing blacklists as follows:

[1]A customer network is a network, which is deploying BLAG for its own *emergency* response.

Network and Distributed Systems Security (NDSS) Symposium 2020
23-26 February 2020, San Diego, CA, USA
ISBN 1-891562-61-4
https://dx.doi.org/10.14722/ndss.2020.24232
www.ndss-symposium.org



(a) Blacklist coverage in routable /24 prefixes that are blacklisted.

(b) Blacklist overlap across different blacklist categories.

(c) IP addresses blacklisted in the same /24 prefix.

Figure 1: Blacklists have low coverage and tend to overlap with other categories of blacklists. Therefore, aggregating blacklists of different types can improve coverage. Also blacklisted addresses tend to be collocated, thus expanding IP addresses to prefixes may further improve malicious source identification.

1) **Aggregation:** BLAG aggregates IP addresses from 157 blacklists of different attack types to improve coverage. BLAG also includes historical listings from these blacklists and assigns relevance score that determines which historical IP addresses are more likely to re-offend.
2) **Estimate misclassifications:** BLAG uses a recommendation system, together with a sample of sources that send inbound traffic to a customer network to tailor its blacklist to this customer. BLAG identifies portions of individual blacklists that may lead to the legitimate source misclassifications and prunes them out. Other portions are aggregated into the master blacklist for this customer.
3) **Selective expansion:** BLAG selects a subset of IP addresses on the master blacklist to expand into /24 prefixes. Only those IP addresses are expanded, where the expansion is not likely to increase legitimate source misclassifications for the given customer.

We present three real-world deployment scenarios for BLAG[2] covering different attacks, where customer networks can reduce the burden on resource-intensive technologies. BLAG improves existing blacklisting approaches by increasing recall (malicious source identification) from 0.1–18.4% to 6.4–69.7%, while maintaining high specificity (legitimate source identification) of 95–99.5%. BLAG also outperforms PRESTA [78], a proposed blacklist aggregation approach by achieving 11.5–84.4% higher specificity, with comparable recall. BLAG also improves the detection delay, discovering malicious sources 8.8–13.4 days faster than competing approaches.

II. Problems With Current Blacklists

In this section, we illustrate the drawbacks that blacklists have, that limit their usefulness in emergency scenarios. We then discuss possible solutions to improve blacklisting and some challenges that we must address. We first show that blacklists generally have low coverage and blacklists of different attack types tend to overlap with one another (Section II-A). This motivates the need for aggregating multiple blacklists of different attack types to improve coverage of malicious sources. We further show IP addresses that were blacklisted in the past get blacklisted again, and sometimes soon after they were removed from a blacklist (Section II-B). This motivates the need for inclusion of historical blacklist data to further improve coverage of malicious sources. Finally, we show that blacklisted IP addresses are often collocated within the same /24 prefix, thus, expanding some IP addresses to prefixes can improve attack detection (Section II-C).

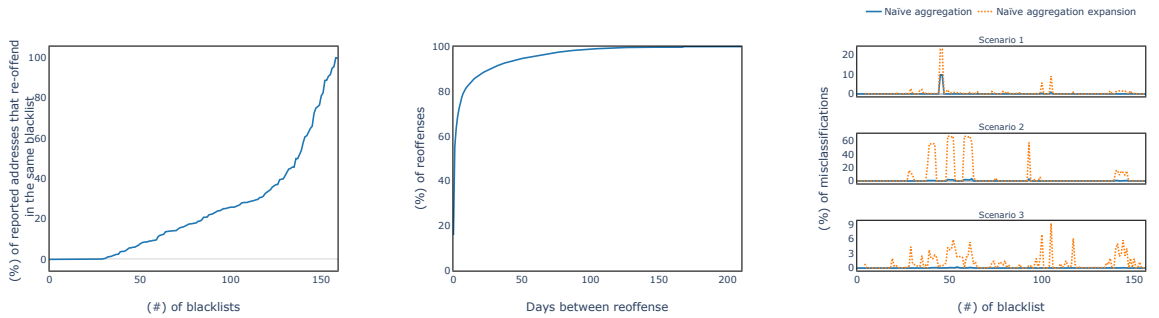
In all cases, some addresses that appear on blacklists may be "wrongly accused", i.e., they may be misclassified, legitimate sources or they may be previously malicious sources, which were since cleaned. We illustrate this in Section II-D to motivate the need for smart, selective aggregation and expansion only of those *portions* of blacklists that are unlikely to contain legitimate sources.

In this section, we leverage our Blacklist dataset, whose details are given in Section IV-A. It consists of 157 publicly available, popular blacklists. We collected their data regularly for 11 months in 2016. We have roughly categorized each blacklist into four categories, based on the type of malicious activities they capture. *Spam* blacklists monitor email spam or emails that contain malicious content and *Malware* blacklists monitor IP addresses that host or distribute malware. *Attack* blacklists, on the other hand, contain IP addresses that initiate DDoS attacks, bruteforce or attacks on specific protocols such as VoIP or SSH. Finally, *Reputation* blacklists list IP addresses that have a low reputation, e.g., because they send unwanted traffic. The algorithm to calculate this reputation is known only to blacklist maintainers.

A. Fragmented Information

Monitoring the entire Internet accurately is impossible. By necessity, each blacklist will gather data from some limited area of the Internet, and thus information about malicious sources will be fragmented over many blacklists. Figure 1(a) illustrates the coverage of individual blacklists in our Blacklist dataset and the unique contribution of blacklists over the dataset "Black24," containing all routable /24 prefixes (extracted from Routeviews [71]) that have at least one blacklisted

[2]Blacklist dataset and code to deploy BLAG can be found at https://steel.isi.edu/Projects/BLAG/



(a) IP addresses in blacklists that re-offend.

(b) Days between reoffense.

(c) Misclassification observed in individual blacklists across three different legitimate addresses.

Figure 2: Blacklisted addresses re-offend quickly. A possible solution is to expand addresses to prefixes, but this causes misclassification of legitimate sources.

address. On average, a blacklist reports only 3.03% of Black24. *Nixspam* blacklist [68] has the highest coverage of 60.7% of Black24. Some blacklists also have unique contributions, i.e., they list addresses from prefixes that appear on no other blacklist. On average, a blacklist contributes unique addresses that belong to 0.16% of Black24. *Nixspam* blacklist has the highest unique contribution – 10.9% of Black24. Previous studies observed similarly low coverage [74], [60] of blacklists.

Another possible reason for fragmented information is the blacklists' focus on a specific type of attacks. This is again by necessity as blacklists built from spam filter alerts will only see spam sources, while intrusion detection systems will only see sources of scans and network attacks. Figure 1(b) shows the overlap of blacklist categories on the y-axis with the blacklist categories on the x-axis. On average, blacklist categories have an overlap of 20.4% with other blacklist categories. The highest overlap is seen in the "attack" category (average 38.7% overlap) and the lowest overlap is seen with the "spam" category (average 2.5% overlap). Although our categorization may not be perfect, we observe that IP addresses are reported across different types of blacklists. Therefore, aggregating multiple blacklists across different attack types can increase blacklist coverage and detect sources of multiple attack types.

B. Re-offense Is Frequent

Figure 2(a) shows the percentage of blacklisted IP addresses (in any blacklist) that have been removed and then appeared again on the same blacklist. On average, 29.3% of blacklisted IP addresses re-offend. Particularly, in two blacklists, *Bambenek Pushdo* and *Palevo*, all IP addresses blacklisted re-offend. However, these are very small blacklists that have reported only 1 and 12 IP addresses during our monitoring period. Figure 2(b) shows the duration between each offense, that is, the number of days the IP addresses stay dormant before they are blacklisted again. On average, reoffenses occur within 9 days and about 91% of reoffense occurs within 30 days. This motivates the need for aggregation of historical blacklist information, especially over the recent past, to improve coverage of malicious sources.

C. Malicious Sources Are Co-located

Prior research has shown that attackers tend to concentrate in a few mismanaged networks [81]. Thus blacklisting an entire network (e.g., one or a set of IP prefixes) could improve coverage of malicious sources. Figure 1(c) shows the number of blacklisted IP addresses in the same /24 prefix, for all /24 prefixes that are present in the blacklist dataset observed during the entire monitoring period. About 57.5% of /24 prefixes have at least two blacklisted IP addresses in the same /24 prefixes. For about 1.7% of /24 prefixes, nearly half the /24 prefix (128 IP addresses) are blacklisted. Only a few /24 prefixes have (0.007%) all their IP addresses in a /24 prefix blacklisted. Identifying prefixes that harbor more malicious sources can increase the blacklist's coverage of malicious sources.

D. Careful Aggregation and Expansion

Not all IP addresses that appear on blacklists are necessarily malicious. Some may have been malicious and got cleaned, which means that the blacklist has stale information. Others may have been misclassified by the blacklisting algorithm. Since blacklists are built using proprietary algorithms, it is impossible to evaluate their accuracy and/or bias.

If we were to naïvely aggregate and expand blacklisted addresses, this would amplify bias and misclassification of legitimate sources. For example, Figure 2(c) illustrates, on the y-axis, the percentage of *legitimate addresses* in our three Scenario datasets (see Section IV for details on datasets) that appears on a blacklist. These address sets are mostly disjoint and are collected from regular inbound traffic in three different networks. The x-axis shows different blacklists in our Blacklist dataset. The order of blacklists is always the same on the x-axis. For this Figure, we used *naïvely aggregated historical data* of each blacklist, i.e, for a given blacklist we produced the list of all addresses that were listed on it up to the time of our Scenario (shown in blue). We make several observations. First, for each Scenario, all misclassifications are concentrated on a few blacklists. On average, about 0.14%, 0.17% and 0.016% of legitimate IP addresses are misclassified. Blacklists such as *Cruzit* [16] and *Chaos Reigns* [4] have high misclassifications of 3.3% and 9.8% respectively. Figure 2(c) also shows that

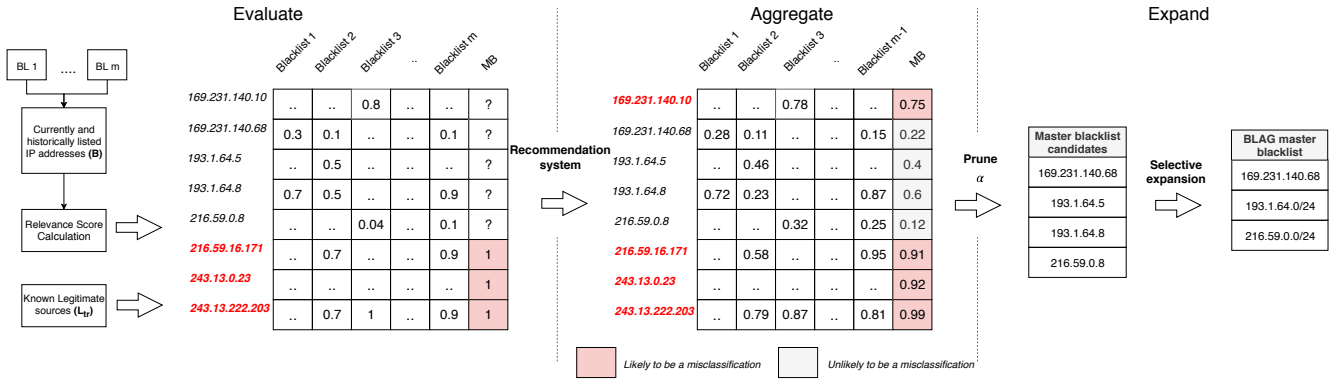


Figure 3: BLAG implementation consists of assigning relevance scores to IP addresses from different blacklists and creating a new *misclassification blacklist (MB)*. The MB contains all possible IP addresses, but a score is only assigned to those that are misclassifications from the known-legitimate sources dataset (L_{tr}). Then, a recommendation system generates missing scores for IP addresses in the MB. IP addresses that have a score higher than α threshold (red blocks in *MB*) are pruned out and the remaining ones are used for aggregation. These IP addresses will be put on a new blacklist known as "Master blacklist candidates". Finally, we selectively expand some candidates into /24 prefixes, if we estimate that this expansion will not severely increase misclassification.

there is no single blacklist that has misclassifications across all three scenarios, thus removing certain blacklists will not solve the problem. Instead, we must *tailor* each blacklist to the customer that is going to use it, to identify and remove portions that may lead to misclassifications of the sources of customer's inbound traffic.

Naïve expansion of blacklisted addresses into prefixes can also lead to misclassifications. We take the naïvely aggregated historical data described earlier, expand every address to its /24 prefix and show the percentage of misclassification in Figure 2(c) (yellow dotted lines). We see that the percentage of misclassifications further increases with naïve expansion. On average, about 0.66%, 6.6% and 1.03% of legitimate addresses are misclassified. Blacklists such as *Cleantalk* [14] and *Chaos Reigns* [4] have high misclassification of 67.2% and 22.6% respectively. Although blacklisted addresses are collocated, naïvely expanding them into /24 prefixes can increase misclassifications.

III. BLAG Design

We present BLAG's design in this section and illustrate the system in Figure 3. BLAG collects historical data from multiple blacklists (B) and updates this dataset whenever a blacklist is updated by its maintainers. When a customer wants to use BLAG, our system uses its historical dataset (B) and a sample of the legitimate sources that send inbound traffic to the customer network (L_{tr}) to curate a master blacklist tailored to that customer.

BLAG's goal when producing the master blacklist is to include as much information about malicious sources from (B) as possible while ensuring that very few current (L_{tr}) or future (L_{te}) legitimate sources of the customer get blacklisted. To achieve this, we need a relatively accurate list of legitimate sources that communicate with the customer. One way a customer could build this list would be to leverage its existing, more sophisticated defenses. Many networks today run an intrusion detection system, a firewall, and a spam filter. These defenses will drop or quarantine some traffic during regular operation. Let us denote the sources of dropped or quarantined traffic as (D) and let (A) represent all sources that have recently sent inbound traffic to the customer. The customer can create (L_{tr}) by starting with a set (A) and removing sources that appear in (D).

BLAG's operation proceeds in the following steps:

(1) *Evaluate* the relevance of every address on every blacklist in (B) (Section III-A), taking into account the listing's age (similar to [78]) and re-offense history. The relevance score is the function of the address's history on a particular blacklist.

(2) *Aggregate* IP addresses and run the recommendation system using IP addresses in the (L_{tr}) set. The system predicts relevance scores of IP addresses that are not in (L_{tr}) but may be among legitimate sources for the customer network in the future ($F \subset L_{te}$). This step ends with a set of addresses that are likely legitimate and likely to communicate with the customer network in the future (F). BLAG then uses a threshold-based approach to prune out current and likely misclassifications (all addresses from (L_{tr}) and most addresses from (F)) and the remaining IP addresses form the master blacklist *candidates*.

(3) *Selectively expand* some candidate IP addresses into prefixes to increase malicious source identification. During this expansion we use (L_{tr}) and (F) sets to estimate the likely increase in misclassification for each candidate if it were to be expanded into an IP prefix. Our expansion method is selective because it balances the gain in malicious source identification against potential future misclassifications (Section III-C).

A. Relevance Scores: Evaluating Quality

Historical blacklist data can be a valuable source to detect potential re-offenders. Earlier, we have seen that about 29% of blacklisted IP addresses re-offend and 91% of these reoffenses

occur within 30 days. We have also seen that blacklists of different attack types overlap. Existing work also agrees with our findings. PRESTA [78], a study on three paid-for blacklists shows that recent offenders are more likely to re-offend. BLAG starts its aggregation by generating a relevance score for each address a listed in blacklist $b \in B$ based on the formulation used by PRESTA. BLAG defines relevance score $r_{a,b}$ as:

$$r_{a,b} = 2^{\frac{l}{t-t_{out}}} \quad (1)$$

where l is a constant, which we set empirically (discussed in Section VII), t_{out} is the most recent de-listing (removal) time of a at blacklist b and t is the time in days when the score is calculated. The exponential factor ensures that the score decays exponentially over time, giving higher weight to recently blacklisted IP addresses. The relevance score ranges from 0 to 1, and a higher score indicates a higher chance of reoffense. If the address a is *currently* listed in b, we set its relevance score to 1.

B. Recommendation System: Estimating Future Misclassifications

Ideally, if we knew all legitimate IP addresses in the Internet at any given time, or if we knew which currently blacklisted addresses are no longer malicious, we could prune out misclassifications during aggregation. However, it is impossible to know this information. At best, a customer network has limited visibility into some set of its known-legitimate sources (L_{tr}), which have recently communicated with the customer. We leverage this set to predict IP addresses (F $\subset$ L_{te}) that may be future legitimate traffic sources for the customer network, and that would be misclassified in BLAG's aggregation and expansion steps.

When all the relevance scores are calculated, BLAG places them into a *score matrix* where blacklists from (B) are at the columns and all listed IP addresses (in any blacklist) are at the rows as shown in Figure 4 (see *Evaluate* step). Each cell in the score matrix holds the relevance score $r_{a,b}$ for the given row (address a) and given column (blacklist b). BLAG adds a new, artificial blacklist to this matrix, called the "misclassification blacklist" (MB column in Figure 4). MB contains all known-legitimate sources from the set (L_{tr}). BLAG assigns a relevance score $r_{a,MB}$ of 1 to all IP addresses a listed in the misclassification blacklist. This high score will help us identify likely future misclassifications (F).

The misclassification blacklist column is sparse because many addresses that exist in (B) do not appear in (L_{tr}) and we cannot know if they are legitimate or malicious. BLAG fills the empty cells of the misclassification blacklist column by using a recommendation system.

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 [59], 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, IP addresses are analogous to movies that are being evaluated, and blacklists are analogous to users assigning the rating. We view the relevance score as the rating.

Two most commonly used types of recommendation systems are content-based recommendation system [69] and collaborative filtering [72]. A content-based recommendation system would require an explicit definition of features that a blacklist uses to determine if an IP address should be listed. Such features are hard to obtain since each blacklist maintainer has its private criteria for listing an address. *Collaborative filtering* infers information about the relationship between a blacklist and an address being listed in a blacklist, by using only the existing relevance scores. That is, it infers the relationship between blacklists and IP addresses, based on when the IP addresses were listed in blacklists and based on similarity in listing dynamics of different blacklists. It then uses the inferred relationships to predict relevance scores for missing cells in the score matrix. We use collaborative filtering in BLAG.

Figure 4 illustrates the recommendation system's operation for a customer network. Let M and N represent the set of IP addresses and blacklists, respectively. Let R be a score matrix of size $|MxN|$ which consists of relevance scores quantifying the relevance 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 IP addresses ($N = 5$). Misclassification blacklist, curated from known-legitimate sources (L_{tr}) for the customer network, is added as the last column in the matrix, and in this example, 128.0.0.5 (highlighted in red) is present in (L_{tr}). Blacklisted IP addresses have been present at various times in different blacklists, which is reflected by the relevance score's value. Address 128.0.0.1 listed in *nixspam* blacklist has a high score of 0.7 since it was the most recently listed address. Address 128.0.0.2, on the other hand, has a low score of 0.1 in *openbl* blacklist, since it was listed long ago. Finally, address 128.0.0.5, has a score of zero in *nixspam* blacklist, where it has never been listed and has a score of 1 in MB since it is known to send legitimate traffic (L_{tr}) to the customer network. There are latent (unknown) features of blacklists and IP addresses that lead to an address being listed in a blacklist. Let the number of latent features that influence relevance scores of IP addresses in blacklists be K (see Section VII for how we choose the value of K). In this example, we set $K = 2$. Our goal is to estimate the relevance scores of IP addresses that are not present in MB, by estimating two matrices $P(|MxK|)$ and $Q(|NxK|)$, which are factors of the original 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:

$$R \approx P \times Q^T = R' \quad (2)$$

We obtain the values of latent matrices P and Q using gradient descent [62], 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 VII 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 IP addresses and latent features K, and each row in Q represents the association strength

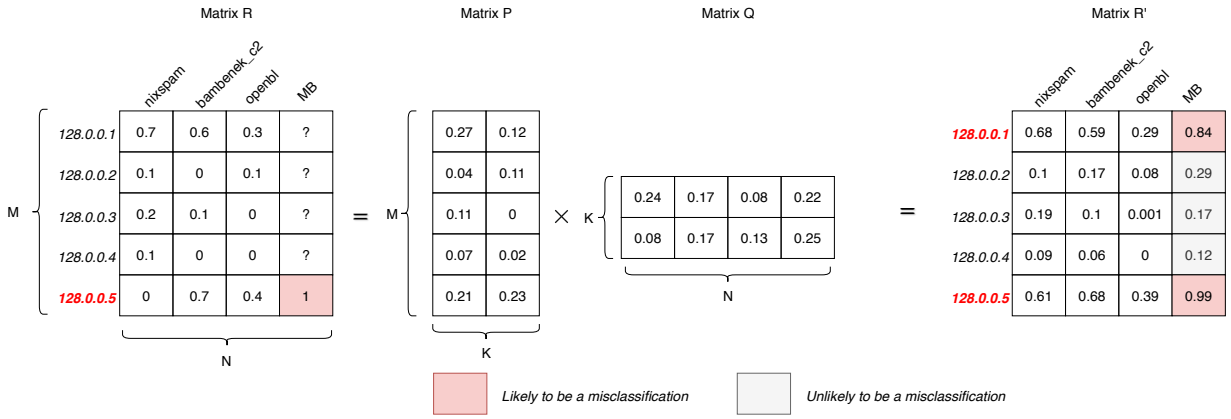


Figure 4: Latent factorization of the score matrix R, a $M \times N$ matrix, where M is the number of IP addresses and N is the number of blacklists. The cells indicate relevance scores. IP addresses not listed in a blacklist are assigned a zero score and IP addresses listed in *misclassification blacklist (MB)* are given a score of 1. 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 missing scores in *MB*.

between blacklists and latent features K. To obtain a relevance score for an address a in misclassification blacklist MB, the dot product of two latent vectors corresponding to address a and MB is calculated as follows:

$$r_{\text{a,b}} = p_a^T q_{MB} \tag{3}$$

Where p_a defines the association strength of address a with features K and q_{MB} defines the association strength of MB with features K.

Consider IP addresses 128.0.0.1 and 128.0.0.5 in Figure 4, where one of them is listed in the MB and the other is not. Both IP addresses have similar relevance scores in other blacklists (with *bambenek_c2*'s scores of 0.6 and 0.7, and *openbl*'s scores of 0.3 and 0.4). Intuitively, if 128.0.0.1 were to be legitimate and listed in MB, we can expect it to have a similar relevance score as that of 128.0.0.5 which is already present in MB. The recommendation system captures this pattern and assigns a score of 0.84 to 128.0.0.1. On the other hand, address 128.0.0.4 has no similar listing pattern as that of 128.0.0.5, therefore the recommendation system assigns it a low score of 0.12 in MB. Finally, IP addresses 128.0.0.2 and 128.0.0.3 share some listings with 128.0.0.5. However, their relevance scores are not similar. This regularity is also captured by the recommendation system and assigns a score of 0.29 and 0.17 respectfully in MB. Cells in the score matrix, in the column MB, that was filled by the recommendation system contain the set of potential future sources of legitimate traffic to the customer (F).

After we have calculated all the missing relevance scores in the misclassification blacklist MB, we proceed to construct the aggregated blacklist known as *master blacklist candidates*. To generate the candidates, we observe relevance scores in MB and then use a threshold α (choice of α discussed in Section VII) to include all the IP addresses a for which the following holds: $r_{a,MB} \leq \alpha$. Intuitively, IP addresses, which have high scores in MB are either current legitimate sources of customer's inbound traffic (L_{tr}) or likely to be so in the future (F), and the thresholding excludes them from the master blacklist.

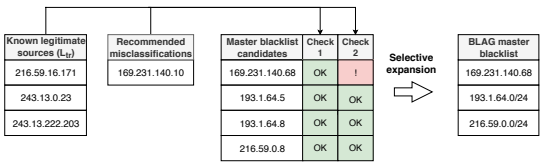


Figure 5: Selective expansion of IP addresses into prefixes.

C. Selective Expansion to Prefixes

We have discussed in Section II-C why it would be useful to identify and expand IP addresses into prefixes. Prior works have expanded IP addresses into prefixes indiscriminately [75], [55], [77] – this improves malicious source identification but greatly increases misclassifications. The novelty of our approach is to *selectively* expand IP addresses into prefixes, only when this expansion does not greatly increase misclassifications. This is particularly useful for customers deploying BLAG under emergency scenarios.

The expansion phase starts with master blacklist candidates, which are all added to the BLAG master blacklist. During expansion, we identify IP addresses, that could be expanded into their /24 prefixes (see Section VI for the rationale behind choosing /24 prefix size). We first generate a list of all /24 prefixes from the master blacklist candidates. We then evaluate if each prefix should be added to the master blacklist. Prefixes that contain known legitimate sources (from (L_{tr})) are excluded (*Check 1*). Further, prefixes that contain likely misclassifications (IP addresses with high relevance scores in the misclassification blacklist, i.e. set (F)) are excluded (*Check 2*). Remaining prefixes are added to the BLAG's master blacklist. In Figure 5, none of the IP addresses are present in known-legitimate sources (L_{tr}) and address 169.231.140.68 has another address in the same /24 prefix, which is a likely misclassification (in set (F)). Therefore, 169.231.140.68 is not expanded, and the other IP addresses are expanded to their corresponding prefix to be included in the master blacklist.

D. Why BLAG Works

BLAG assigns relevance scores to capture the relevance of IP addresses being listed in a blacklist. BLAG also introduces a new artificial blacklist – misclassification blacklist, which consists of known-legitimate sources (L_{tr}). The recommendation system used by BLAG helps during the aggregation phase by pruning out misclassifications, and also during the selective expansion phase by preventing those expansions of IP addresses into prefixes that would increase future misclassifications. The recommendation system helps BLAG to discover other IP addresses that are predicted to be listed in the misclassification blacklist with a high relevance score. In other words, the recommendation system predicts future misclassifications based on finding IP addresses that exhibit similarities, with regard to the blacklisting process, to known legitimate sources. In Section VI-B, we quantify the contribution of the recommendation system in reducing misclassification during the aggregation and selective expansion phases.

IV. Datasets

BLAG's fundamental goal is to find a trade-off between identifying as many malicious sources as possible and keeping the misclassifications low. In this section, we look into the blacklists used by BLAG to aggregate information. We also present three BLAG deployment scenarios, which we use in evaluation. These scenarios include real-world legitimate and malicious traffic. We will show the performance of BLAG under these scenarios in Section V, where BLAG achieves more than 95% specificity (5% misclassification rate), while significantly increasing recall (high detection of malicious sources), compared to individual blacklists and their naïve aggregation.

A. Blacklist Dataset

We have monitored 157 publicly available blacklists for 11 months, starting from January 2016 to November 2016. Each blacklist is updated at a different frequency by its provider, ranging from 15 minutes to 7 days. We collected the update time of each blacklist manually and programmed our crawler to pull the snapshot of the blacklist when a new update was available. We have collected around 176 million blacklisted IP addresses over 23,483 autonomous systems. Our monitored blacklists vary in size – on one hand, we have large blacklists (15.76%) listing more than 500,000 IP addresses and on the other, we have small blacklists (19.56%), which list fewer than 1,000 IP addresses. We do not delve into details on the various properties of blacklists, such as their volume, contribution, exclusive contribution, detection of malicious activities (refer Vector et al. [63] work on an exhaustive study on properties of blacklists). Our work focuses more on identifying key properties of blacklists that make them ineffective in emergency scenarios (refer Section II) and presenting an improved blacklisting technique (refer Section III).

Our blacklist dataset (B) is representative of different attack vectors such as spam, malware, DDoS attacks, ransomware, etc. Table II shows the blacklist maintainers and the number of blacklists maintained by them. Our dataset includes popular blacklists such as DShield [57], Nixspam [68], Spamhaus [44], Alienvault [3], Project Honeypot [37], Abuse.ch [47] and Emerging Threats [21].

B. Scenarios

Table I shows our three scenarios. Each scenario consists of three portions of the same dataset: training, validation and testing. The training portion contains only known-legitimate sources (L_{tr}) and is used to tailor BLAG to the customer network (Section III-B). This portion is collected before the malicious event in each scenario. The validation and testing portions contain both the legitimate (L_v and L_{te}) and malicious (M_v and M_{te}) sources. The validation portion is used to calibrate BLAG's parameters (l, α and K) for testing[3]. The testing portion is used to evaluate the performance of BLAG and competing blacklisting approaches.

Our three scenarios contain sources of diverse attacks: spam, DDoS on a University network or DDoS on DNS root. This allows us to test how well BLAG could prevent these attacks if used by a customer network to filter attack traffic.

*1) **Malicious Email Campaign or Spam**:* In this scenario (referred to as **Email**), we look into a case where a University network is bombarded with spam emails. We collect malicious and legitimate IP addresses during the same time period of June 2016. Simultaneous collection is important, because an address may be malicious at one time, and cleaned afterward. We collect malicious IP addresses 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 SpamAsssassin [64] to obtain around 2.3 M spam e-mails, sent by around 39 K IP addresses. These IP addresses form our malicious dataset. We trained SpamAssassin using SpamAssasin's public corpus [43] and spam archives from Untroubled [49], to ensure we capture only malicious spam emails. We use the first 7 days consisting of 13 K IP addresses as a validation set (M_v) and the remaining 16 days consisting of 26 K IP addresses for testing (M_{te}). We collect legitimate IP addresses through a human user study. This study was reviewed and approved by our IRB. We recruited 37 volunteers from our University, 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 the OAuth2 protocol to access Gmail, and it used regular expressions to extract a sender's IP address, time and label for each e-mail. We did not extract any other information and did not record the identity of the study participants, to protect privacy. 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 extracted around 30 K e-mail records, sent by around 9 K IP addresses. We use the first seven days of this dataset consisting of 3 K IP addresses for training (the known-legitimate sources set (L_{tr})), the next 7 days consisting of 2 K IP addresses for validation (L_v) and the remaining 16 days consisting of 4 K IP addresses for testing (L_{te}).

*2) **DDoS on a University network**:* In this scenario (referred to as **$\text{DDoS}_{\text{Univ}}$**), we look into a case where webservers at a University could be targeted by Mirai-infected

[3]We do not have a validation dataset for DDoS_{DNS}. Refer Section VII for further explanation

Scenario	Duration	Source	Training (L_{tr})		Validation (L_v+M_v)		Testing (L_{te}+M_{te})	
			Days	IPs	Days	IPs	Days	IPs
Email	6/1/16-6/30/16	M:Mailinator	-	-	7	M_v: 13 K	16	M_{te}:26 K
		L: Ham	7	L_{tr}: 3 K	7	L_v: 2 K	16	L_{te}: 4 K
$DDoS_{Univ}$	9/1/16-9/30/16	M:Mirai	-	-	7	M_v: 1.1 M	16	M_{te}: 2.8 M
		L: Web cl.	7	L_{tr}: 16K	7	L_v: 12 K	16	L_{te}: 33 K
$DDoS_{DNS}$	6/24/16-6/25/16	M:DDoS	-	-	-	-	1	M_{te}: 5.5 M
		L: DNS cl.	1	L_{tr}: 2.7M	-	-	1	L_{te}: 16 K

Table I: Scenario datasets used in this study. Each scenario dataset is split into three – training, validation and testing. The training dataset is collected chronologically before the validation and testing, and contains only legitimate sources (L_{tr}). The validation and testing datasets are collected during malicious events and contain malicious (M_v and M_{tr}) and legitimate (L_v and L_{tr}) sources. The validation dataset is used to tune the appropriate parameters used in BLAG.

Type	#	Blacklist Maintainers
Malware	57	Emerging threats [21], Malware Bytes [25], CyberCrime [17], URLVir [50], Swiss security blog [47], Bambenek [6], NoThink [34], I-Blocklist [26], NoVirusThanks [35], DYN [20], Malc0de [30], Malware domain list [31], Botscout [9], ImproWare [27]
Reputation	32	Emerging threats [21], Graphiclineweb [23], Alienvault [3], Binary Defense Systems [7], CINSscore [11], Swiss Security Blog [47], Blocklist.de [8], I-Blocklist [26], Cisco Talos [12], Bad IPs [5], Blocklist Project [22], VXVault [52], ShunList [41], GreenSnow [24]
Spam	39	Spamhaus drop and edrop [44], Stop Forum Spam [46], Chaosreigns [4], Lashback [28], Nixspam [68], Project Honeypot [37], Sblam! [40], Turris [38], ImproWare [27], Malware bytes [25], Cleantalk [14], My IP [33], BadIPs [5]
Attacks	29	I-Blocklist [26], Malware Bytes [25], Snort Labs [42], TrustedSec [48], Haley [10], Darklist [19], SIP blacklist [45], VoIPBL [51], DShield [57], NoThink [34], OpenBL [36], Cruzit [16], BruteforceBlocker [18], Clean MX [13], Bad IPs [5], MaxMind [32]

Table II: 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.

hosts. The legitimate and malicious IP addresses are collected during September 2016. We collect malicious IP addresses from Netlab's [2] collection of Mirai-infected hosts during September 2016. There were around 3.9 M addresses, which form our malicious set. We use the first 7 days consisting of 1.1 M IP addresses as a validation set (M_v) and the remaining 16 days consisting of 2.8 M IP addresses for testing (M_{te}). We collect legitimate IP addresses by identifying Web clients who communicated with a set of popular web servers at a mid-size US University in September 2016. We included only those TCP connections which exchanged payload with the server (thus excluding scans). This resulted in around 61 K legitimate IP addresses. We use the first 7 days consisting of 16 K IP addresses as the known-legitimate sources (L_{tr}), the next 7 days consisting of 12 K IP addresses for validation (L_v) and the remaining 16 days consisting of 33 K IP addresses as the future legitimate sources (L_{te}).

*3) **DDoS on DNS root**:* In this scenario (referred to as **$DDoS_{DNS}$**), we look into a case of TCP SYN flood attack on the DNS B-root server [58]. We communicated with the dataset provider to obtain non-anonymized sources of this attack, as well as non-anonymized data before the attack, which we use for training. We mine the malicious IP addresses (M_{te}) as those that have sent TCP SYN flood to the server for two hours on June 25, 2016. There are 5.5 M malicious IP addresses. We mine the known-legitimate sources (L_{tr}) as sources of DNS queries 1-day before the attack event (2.7 M IP addresses), and the future legitimate sources (L_{te}) as sources of DNS queries during the attack event (16 K IP addresses).

C. Limitations

The blacklist dataset contains only publicly available blacklists, while many providers also offer for-pay blacklists that are usually larger and more accurate [78] (see Section V for the performance of our monitored blacklists). We chose to use only publicly available blacklists because: (1) these blacklists are widely used and we wanted to evaluate BLAG's benefits for a customer network that deploys such blacklists. A recent survey shows that nearly 60% of surveyed network operators use blacklists including publicly available ones [56]. We also believe that BLAG's benefits would carry over to for-pay datasets because its mechanism is generic. (2) we wanted our work to be repeatable, and using public blacklists enables us to freely share our data. We plan to share the blacklist dataset and BLAG code, which could be used by any customer network to improve blacklisting.

Our scenario datasets suffer from several limitations. First, they only capture a small sample of legitimate/malicious IP addresses that were active on the Internet at a given point in time, and for a given legitimate or malicious purpose. Many other IP addresses could be legitimate or malicious at the same time, and we have no ground truth for these. We also rely on other security technologies (SpamAssassin) that may also be used by blacklist maintainers to blacklist an address. These limitations are present in other published works [80], [70], [75], [78], [55], which use similarly-sized malicious and legitimate datasets, and rely on other security technologies such as Proofpoint [39] and SpamAssassin, as we do, to establish maliciousness at a given time. A more recent study on Blacklisting has used Alexa's top 10,000 list to evaluate the accuracy of blacklists based on IP addresses that are present in them [63] but Alexa's rankings are also not ideal measure of legitimacy [73], [61]. These limitations cannot be avoided, as there is no complete, 100% accurate list of legitimate and malicious IP addresses on the Internet nor in any specific network, at any given point in time.

Second, our datasets are dated – captured in 2016. Ideally, we would use more recent datasets. But, it is very hard to find data about attack events and legitimate traffic, which includes non-anonymized IP addresses. It is even harder to find such data streams that are collected simultaneously and that relate to the same type of sources (e.g., sources of legitimate email vs sources of spam). While dated, our scenarios faithfully capture sources of legitimate and malicious traffic at the same time.

When working with each scenario *BLAG only uses blacklist data up to that point in time*. Thus we faithfully simulate what BLAG's performance would have been if it were deployed by a customer network at the time.

Finally, we do not have a validation dataset for the *DDoS$_{DNS}$*, because the attack on B-Root DNS server was observed only for a few hours. From evaluation of our parameter values on Email and DDoS$_{\text{Univ}}$ datasets (Section VII) we observe that parameters l (historical decay) and K (matrix factorization) do not change with the dataset, and high values of parameter α (threshold for candidate IP addresses) lead to higher specificity. Therefore, to evaluate the performance of BLAG for DDoS$_{\text{DNS}}$ scenario, we use the same l and K parameters as that of Email and DDoS$_{\text{Univ}}$ scenarios and set a high value for α.

V. Evaluation

In this section, we evaluate the performance of BLAG and several competing approaches, for different deployment scenarios, described in Section IV. We find that BLAG achieves high specificity (95–99%), and has much better recall (3.5x–114x improvement) than competing approaches. BLAG also detects attackers 13.7 days faster than competing approaches.

A. Evaluation Setup

We measure performance of blacklists using *recall* and *specificity*. Recall measures the percentage of malicious sources (out of some ground-truth set) that were blacklisted. Specificity measures the percentage of legitimate sources (out of some ground-truth set) that were not blacklisted. Higher values for both measures denote better, more accurate, blacklists.

In this section, we compare several competing blacklisting approaches against BLAG. In all cases when we evaluate a blacklisting approach we "travel back in time" to the time just before the testing portion of our given Scenario. We then use past data from our (B) dataset up to the time when the testing portion starts, and refresh it as blacklists update during the testing. For example, imagine if our scenario contained two days Sep 1st, 2016 and Sep 2nd, 2016, with Sep 1st used for training and Sep 2nd for testing. Our blacklist dataset (B) overlaps this period going from Jan 8th, 2016 to Nov 30th, 2016. When we test a blacklisting approach for the given scenario we would start by including all data from (B) from Jan 8th up to, and including Sep 1st. We would then start our evaluation and keep updating the blacklist on Sep 2nd. This way we faithfully simulate the performance a given approach would have if we were to travel back in time to Sep 2nd.

We compare the performance of the following approaches against BLAG:

Best – the blacklist from Blacklist dataset that performs the best on a given scenario with regard to recall. *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 the benefits of aggregation over the use of a single blacklist.

Historical – all IP addresses listed in any blacklist in the Blacklist dataset. This approach assumes "once malicious, always malicious" and performs naïve aggregation.

PRESTA+L – the blacklist generated using techniques described in [78]. PRESTA performs spatio-temporal analysis and expansion using historical blacklist data to generate a more proactive blacklist. PRESTA assigns a reputation score to all blacklisted IP addresses across three different spatial groups (IP address, address's /24 prefixes along with two surrounding /24 prefixes and address's corresponding autonomous systems). PRESTA combines all the spatial grouping into one and chooses relevant listings using a thresholding technique. We tune PRESTA such that the BLAG's recall is equivalent to that of PRESTA for a deployment scenario, and then we compare the specificity of these two approaches. Further, to tease apart the factors that help us outperform PRESTA, we consider a variant approach, called *PRESTA+L*, where we remove addresses that are already present in known-legitimate sources (L_{tr}) from every dataset.

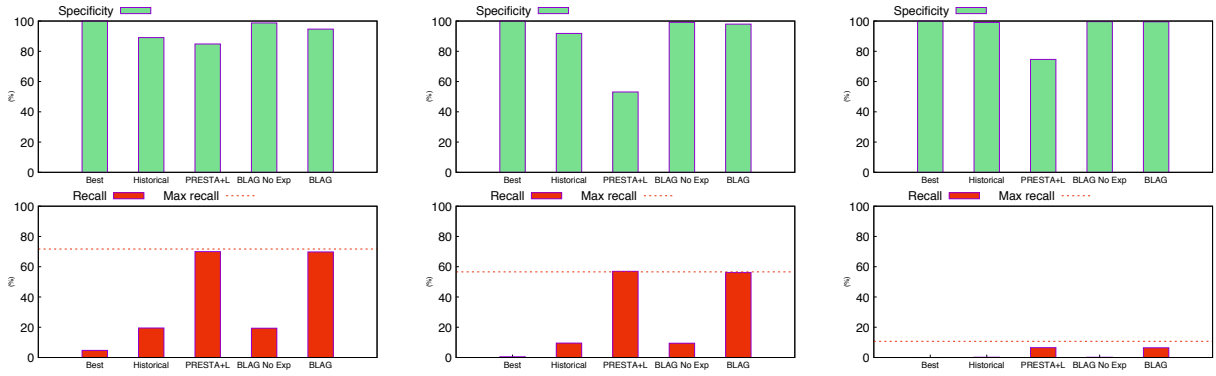
BLAG with and without selective expansion: We run BLAG as described in Section III. We set the length of address history $l = 30$ and the number of latent features for recommendation system $K = 5$. We set relevance threshold $\alpha = 0.8$ for Email/DDos$_{\text{DNS}}$ datasets and $\alpha = 0.6$ for DDoS$_{\text{Univ}}$ dataset. Our choices for these variables are explained in Section VII. We compare the performance of BLAG with and without the selective expansion, to show the contributions of aggregation and expansion stages of BLAG.

During the evaluation, for our testing dataset and each blacklisting approach (best, historical, PRESTA+L or BLAG) we simulate the dynamics of address appearance over time in the following manner. When an address appears in the blacklist dataset (B) we: (1) include the address in the best blacklist if it appeared on a blacklist, which will ultimately perform the best on the given BLAG deployment scenario, (2) include the address in the historical blacklist, (3) apply PRESTA+L algorithm on the address and all its spatial groups to determine if it is included in the PRESTA+L blacklist, (4) 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. We report recall and specificity at the end of the testing datasets.

B. BLAG is More Accurate

BLAG's goal is to capture as many malicious sources as possible while keeping the specificity high.

BLAG has the best specificity/recall tradeoff across the three scenarios: Figure 6 shows that BLAG overall has the best performance. For the Email scenario, the best blacklist has higher specificity (100%) than BLAG (95%). However, BLAG improves the best blacklist's recall from 4.7% to 69.7%. Historical blacklist has a recall of 19.4%, which is better than the best blacklist but not better than BLAG. Naïve aggregation of IP addresses in historical blacklist lowers its specificity to 89% which is much lower than that of BLAG (95%). For the same recall, BLAG has 11.5% better specificity than PRESTA+L. We observe similar pattern with DDoS$_{\text{Univ}}$ and DDoS$_{\text{DNS}}$ scenarios. BLAG's recall ranges between 6.4–56.1% when compared to 0.004-0.4% for best blacklist and 1.8–9.5% for historical blacklists. For the same recall, BLAG's specificity ranges from 97.9–99.5% when compared to 53.1–84.8% of PRESTA+L. We detail in Section VI-B how the



(a) Email (b) $DDoS_{Univ}$ (c) $DDoS_{DNS}$

Figure 6: Specificity and recall of BLAG with two competing approaches on traffic datasets.

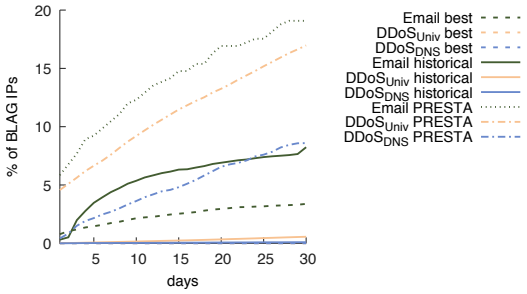


Figure 7: Delay in reporting malicious IP addresses reported by BLAG.

contribution of recommendation systems in BLAG helps it achieve better specificity that PRESTA+L.

BLAG filters more attack traffic. Some IP addresses may generate more attacks than others. We evaluate the amount of malicious activity that would be filtered by BLAG, best and historical blacklists for our three scenarios. In the Email scenario, BLAG would filter 69.7% of spam, compared to 4.7% and 19.4% filtered by best and historical blacklists respectively. In the $DDoS_{Univ}$ scenario, BLAG would filter traffic from 56.1% of infected devices, compared to only 0.4% and 9.5% filtered by best and historical blacklists. In the $DDoS_{DNS}$ scenario, BLAG would drop 6.4% of attack queries, compared to 0.004% and 0.1% filtered by best and historical blacklists.

We pause here to address the low performance of public blacklists in general for offender identification. While BLAG greatly improves performance, it can only work with what it has – existing public blacklist data. For all malicious datasets, blacklists cover addresses in only 10.7–71.7% of /24 address spaces and BLAG's recall cannot go beyond this (shown by the red horizontal line in Figure 6). Overall BLAG manages to identify 6.4–69.7% of attackers and drop similar amounts of attack traffic. While this is far from perfect, dropping more than half of the attack traffic may be very helpful in emergency scenarios. If BLAG were used to aggregate for-pay blacklists, which list many more malicious sources [78], its attacker coverage would likely be better. We emphasize, however, that BLAG manages to greatly improve the performance of public blacklists while limiting collateral damage to legitimate traffic.

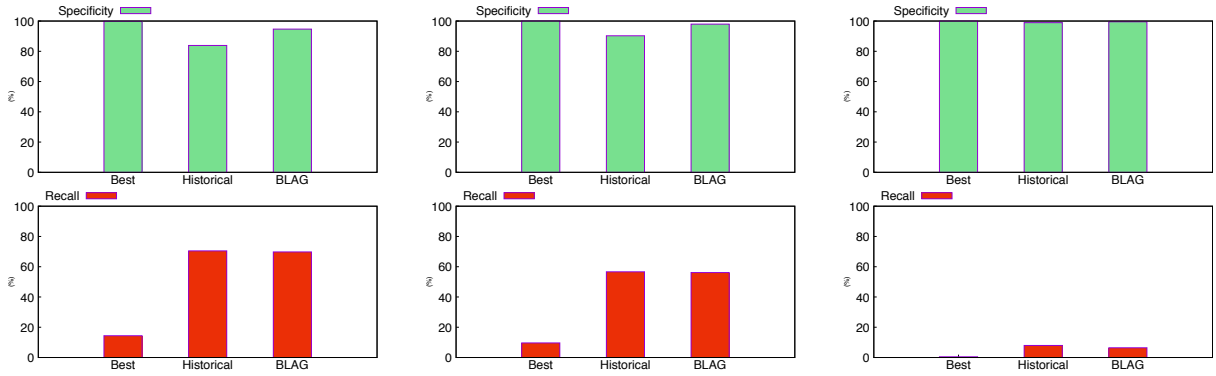
BLAG lists malicious sources faster. We show in Figure 7 the number of days taken by other competing approaches to list malicious IP addresses after BLAG discovers them. We track competing approaches for up to 30 days. On average across all scenarios, BLAG reports malicious sources 9.4 days faster than the best blacklist, 10.3–16.1 days faster than historical blacklists and 8.8–13.4 days faster than PRESTA+L. BLAG's aggregation helps in detecting attackers faster than the best blacklist and BLAG's selective expansion helps in detecting attackers faster than the historical blacklist. After 30 days, the best blacklist catches up to 2.1% of malicious IP addresses discovered by BLAG for the Email scenario and does not report any malicious address for $DDoS_{Univ}$ and $DDoS_{DNS}$ scenarios. On the other hand, historical blacklists catch up to 0.2–4.1% and PRESTA+L catch up to 0.14–0.23% of malicious IP addresses discovered by BLAG.

VI. Sensitivity Analysis

In this section, we discuss the contribution of BLAG's expansion phase. We then look into the contribution of BLAG's recommendation system in pruning misclassifications for the aggregation and expansion phase.

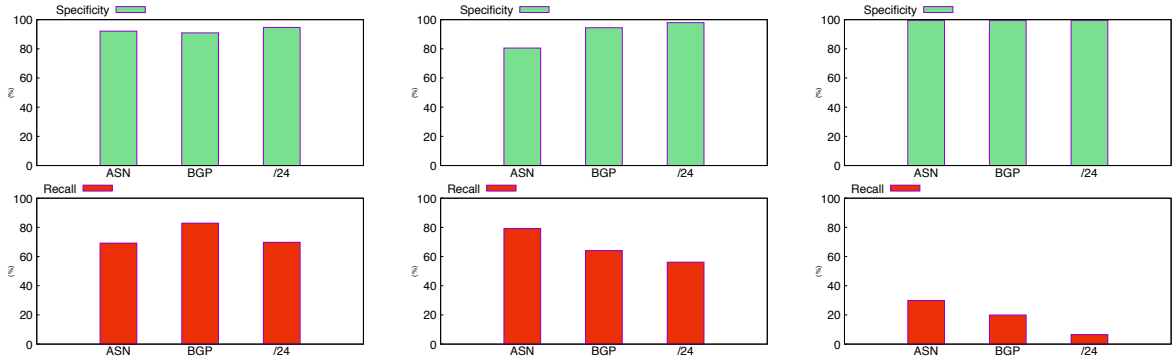
A. Expansion approaches

BLAG's performance comes both from aggregation and expansion. We investigate 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 6). Without expansion, for the Email scenario, BLAG's recall is 19.3% and its specificity is 98.7%, while the best blacklist has 8.9% recall and 100% specificity. BLAG is thus still better than the best blacklist, even without expansion. The historical blacklist has 19.4% recall (vs 19.3% of BLAG without expansion), but it has 89% specificity (vs 98.7% of BLAG). Finally, PRESTA+L has 68% recall (vs 19.3% of BLAG without expansion), but it has much lower specificity (84.8% vs 98.7%). This is expected since PRESTA+L applies expansion and we compare it to BLAG without expansion. Expansion of BLAG improves recall further (to 69.7%), at a small loss of specificity (95% specificity of BLAG with



(a) Email (b) $DDoS_{Univ}$ (c) $DDoS_{DNS}$

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



(a) Email (b) $DDoS_{Univ}$ (c) $DDoS_{DNS}$

Figure 9: Evaluating BGP and AS expansion techniques.

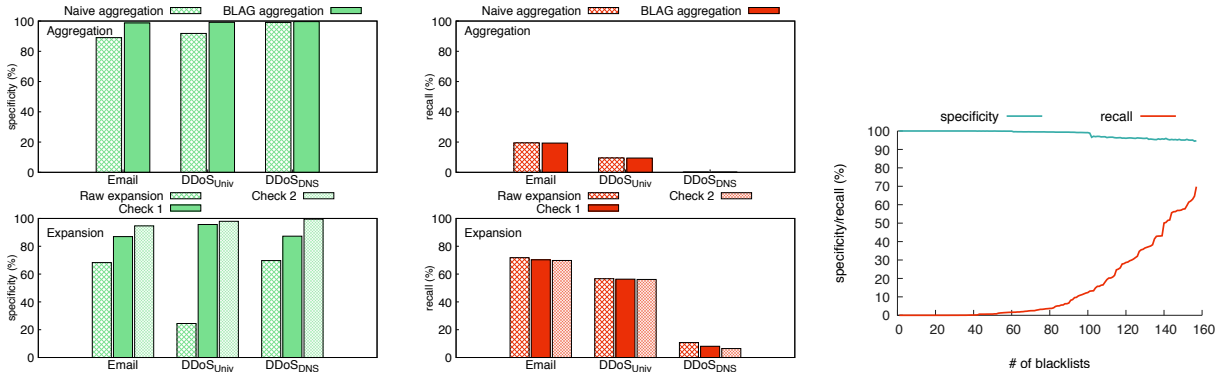
expansion vs 84.8% of PRESTA+L), making BLAG the best performing approach, among the ones we tested. Similar trends hold for other scenarios we tested.

BLAG outperforms best-expanded and historical-expanded blacklists. We investigate if a similar expansion approach to BLAG's could improve the performance of the best blacklist and the historical blacklist. Instead of selective expansion here, which uses BLAG's information, we use naïve expansion, where each candidate address is expanded into its /24 prefix if there are no overlaps with the known-legitimate source (T_{tr}) dataset. Figure 8 shows that BLAG still outperforms competing approaches, due to its selection of only high-quality information to aggregate, before expansion. For the Email scenario, the best blacklist with expansion has 99.8% specificity vs 95% of BLAG. But, BLAG has 69.7% recall, while the best blacklist with expansion has only 14.4%. The historical blacklist with expansion has a comparable recall to that of BLAG (69.7% vs 70.4% respectively). However, BLAG has 95% specificity while the historical blacklist with expansion has 83.8%. We observe similar trends in the $DDoS_{Univ}$ and $DDoS_{DNS}$ scenarios. In the $DDoS_{DNS}$ scenario, the historical blacklist with expansion achieves 98.9% specificity (vs 99.5% of BLAG) and 7.9% recall (vs 6.4% of BLAG) so they perform roughly equal.

BLAG's expansion approach using BGP prefix and AS level. We investigate how BLAG's performance would change if we expanded IP addresses into their full BGP prefixes or entire autonomous systems, as suggested in [75]. Expanding some IP addresses into large prefixes is not advisable, as this can potentially have large collateral damage, but we investigate it as a hypothetical scenario. We apply /24-prefix, BGP-prefix and AS-level aggregation to the master blacklist candidates, produced by BLAG for the three datasets. We then apply the selective expansion technique, but instead of using /24 prefix in deciding whether to expand, use the encompassing BGP prefix and the entire AS. We show the specificity and recall of BLAG with these expansion approaches in Figure 9. Overall, expanding IP addresses to AS or BGP prefix has a higher recall than /24 expansion (7.9–23.4% higher). However, specificity varies across different deployment scenarios. For the Email scenario, /24 expansion has a little better specificity than AS and BGP prefix expansion (2.5–3.7% better). For the $DDoS_{Univ}$ scenario, /24 expansion has better specificity than AS and BGP prefix hijacking (3.4–17.4%). For the DNS scenario, specificity is about the same across the three expansion approaches.

B. Contribution of recommendation system

Figure 10(a) shows the contribution of the recommendation system in the aggregation and selective expansion phase over



(a) BLAG's recommendation system contribution to specificity.

(b) BLAG's recommendation system contribution to recall.

(c) Contribution of blacklists by size.

Figure 10: Evaluating impact on specificty/recall due to various components in BLAG and contribution of blacklist size to BLAG.

our three scenarios. During the aggregation phase (top of Figure 10(a)), BLAG's recommendation system can prune out misclassifications and improve specificity from 89–99% to 98–99.6%. BLAG is even more effective in increasing specificity during the selective expansion phase. BLAG's recommendation system (shown as $check1$ in bottom half of Figure 10(a)), improves specificity from 24.5–69.7% to 86.9–95.6% over naïve expansion of all IP addresses. Also, BLAG's complete selective expansion phase (shown as $check2$), further improves specificity to 95–99.5%.

On the other hand, BLAG's recall depends on the coverage of blacklist, aggregation and selective expansion phases. Figure 10(b) shows the impact of BLAG on recall. BLAG's aggregation phase (top of Figure 10(b)) uses a threshold to prune out false positives and this can also prune out malicious addresses. This phase reduces recall from 0.18–19.4% to 0.15–19.35%. BLAG's selective expansion (bottom of Figure 10(b)) also reduces the recall. During *check1* phase of selective expansion, recall further from 10.7–71.9% to 9–70.2% and during the *check2* phase of selective expansion, the recall further reduces to 6.4–69.7%.

C. Contribution of Individual Blacklists

We ran BLAG on n largest blacklists for the Email scenario, and varied n from 1 to 157 as shown in Figure 10(c). There is a 15.7% gain in recall for the first 106 blacklists. After this, there is a sharp increase in recall for the remaining blacklists. The next 20 largest blacklists double the recall to 32% and the next 14 largest blacklists push the recall past the 50% mark. When including all blacklists, the recall reaches 69.7%. Specificity stays at 100% for the first 41 blacklists and then drops slightly from 100% to 99.1% for the first 100 blacklists. The specificity drops to 95.5% for the next 40 blacklists and finally, by adding the remaining blacklists, the specificity comes down to 95%. Because we see a steady improvement in recall and a steady, slow decline in specificity, it is hard to tell how many blacklists would be enough. Instead, BLAG could let a customer choose different numbers of blacklists to aggregate and could produce specificity estimates (e.g., by using a validation set L_v) for every choice.

D. Contribution of known-legitimate sources (L_{tr}):

We ran BLAG by varying the duration of the (L_{tr}) available to BLAG– 0, 1, 3, 5 and 7 days for Email dataset as shown in Figure 11(a). As the number of days is increased, the specificity improves with small reduction in recall. With only 1 day of legitimate data available, the specificity improves by 36.7% with a loss of 2% of recall. As we increase the number of days the gain in specificity increases. By including all the days for training dataset, the specificity only improves by 4.5% and the recall reduces by 4.7%.

VII. Parameter Tuning

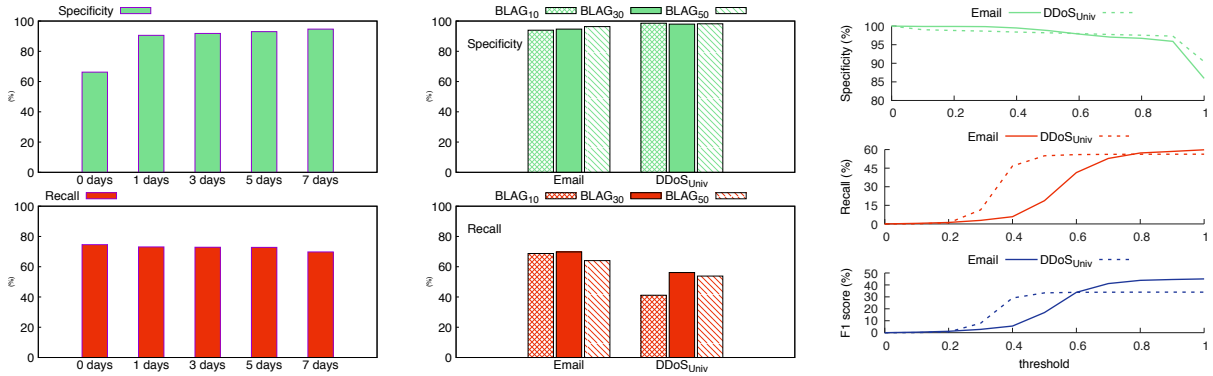
In this section, we discuss a methodology used to determine the parameters l, α and K[4].

Parameter l for Historical Decay. In Section III-A, l roughly controls the length of historical data (in days) that may be included in the BLAG master blacklist. Using the validation dataset (L_v + M_v), we determine the impact of l for three values (10, 30 and 50) shown in Figure 11(b). For the two scenarios, $l = 30$ has the highest recall. About 1.5–8.9% and 4.3–36.4% higher recall for Email and $DDoS_{Univ}$ scenarios respectively. For $l = 30$, the specificity is uniformly high, ranging from from 95–97.9%. Therefore, $l = 30$ strikes the right balance, which increases recall with minimal loss in specificity. During our evaluation, we use $l = 30$ for the $DDoS_{DNS}$ dataset. This agrees with our observations in Section II-B, where 91% of blacklisted addresses that re-offend, do so within 30 days.

Parameter α for choosing Master Blacklist candidates. The parameter α controls the set of IP addresses, which should be considered for the expansion phase in BLAG. We show the performance in the validation dataset (L_v + M_v). Figure 11(c) shows that for each scenario parameter α trades accuracy of BLAG with higher coverage. For various thresholds, we plot the specificity, recall and F1-score[5]. Network operators could

[4]Scripts to generate l, α and K can be found at https://steel.isi.edu/Projects/BLAG/

[5]F1-score is the harmonic mean of precision (fraction of IP addresses, which are listed and are indeed malicious sources) and recall



(a) Contribution of known-legitimate sources dataset.

(b) Evaluating parameter l.

(c) Threshold α.

Figure 11: Sensitivity analysis

use the F1-score as a metric to determine the appropriate threshold. We see that for the Email scenario, after a threshold of 0.8, the F1-score plateaus at 43%. For higher thresholds, the specificity drops from 95% to 85.9%. Similarly for $\text{DDoS}_{\text{Univ}}$ scenario, the F1-score plateaus after a threshold of 0.6 at 33%. For higher thresholds, the specificity again drops from 97.9% to 90.4%. Therefore, we set a threshold of 0.8 for Email and 0.6 for the $\text{DDoS}_{\text{Univ}}$. Relevance scores for misclassifications would typically be high since all misclassifications are allocated a score of 1 in the misclassification blacklist. Therefore, for DDoS_{DNS} scenario, we set a threshold of 0.8 to prune out misclassifications.

Parameter K for Matrix Factorization. Parameter K is used in non-negative matrix factorization (NMF), and denotes the number of latent features. An ideal K will have the minimum error between the matrix $R(MxN)$ and R' (Section III-B). Brunet et al. [53] suggested using the smallest K, after which the *cophenetic correlation coefficient* starts decreasing. For the validation datasets, we evaluate different values of K and find that the cophenetic correlation coefficient starts decreasing after K is 5. BLAG is pre-configured to run gradient descent with $K = 5$ until the root mean squared error (RMSE) between the original matrix R and matrix R' fell below 1% or the number of iterations exceeded 1,000.

VIII. Related Work

In this section, we survey work related to blacklist analysis, blacklist improvement and 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 IP addresses, and active malware domains [60]. 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. work support our findings that blacklists are not accurate. Zhang et al. evaluated nine blacklists using traffic logs from a regional ISP [80]. They analyzed overlapping IP 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. Vector et al. [63] analyzed 47 distinct IP address threat intelligence sources and evaluated them for volume, differential contribution, exclusive contribution, latency, coverage, and accuracy. They used Alexa top 10 K websites as ground truth for legitimate sources and scanners captured by CAIDA darknet as ground truth for malicious sources. Vector et al. support our finding that blacklists or threat intelligence feeds have low recall ($< 2\%$) and low misclassification as well. The main difference between these related works and ours is twofold. First, our main focus is on distilling accurate pieces of information from blacklists and aggregating it into a master blacklist; we use data about current blacklists merely to motivate our work. Second, we use an order of magnitude more blacklists than previous works.

Improving Blacklisting. PRESTA [78] uses historical data from three pay-for spam blacklists provided by Spamhaus [44] to infer temporal and spatial properties of IP addresses and expand IP addresses into three spatial regions. Spamhaus blacklists are highly reputable and have high recall ($> 80\%$) and very low misclassifications ($< 2\%$). PRESTA's technique helps to uncover 18% more spammers while keeping misclassifications low. In our evaluation, too, PRESTA helped improve recall but at much higher misclassification cost (Section V). This may be because we use public blacklists, which may not as accurate as pay-for blacklists.

Highly Predictive Blacklisting [79] (HPB) creates blacklists customized to a given network by a page ranking algorithm. Soldo et al. [76] extended HPB to use historical data about attack sources and destinations and used a recommendation system to predict possible attackers for each victim. In contrast, BLAG uses a recommendation system to remove misclassifications while aggregating blacklists. Sinha et al. [75] present a new spam blacklisting technique, by monitoring email servers and spam traps to curate a spam blacklist. BLAG, on the other hand, works with existing blacklists to improve their performance and is applicable to different types of attacks.

IX. Discussion

We discuss possible attacks on BLAG. BLAG has no way, other than the recommendation 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, which does not have much overlap with MB (e.g., by copying it) and if they 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. Networks today choose carefully which blacklists to use, based on their reputation in the operator community. We assume they would apply the same care to choose blacklists used by BLAG. BLAG makes any polluted information less likely to propagate than individual blacklists. The attacker would have to carefully craft the polluted blacklist so that the servers appear on the same blacklist as many malicious sources, and their appearances do not resemble appearances of the known-legitimate sources on MB. Otherwise, BLAG would be able to identify and discard low-quality information. We leave the exact handling of pollution attempts for our future work.

X. 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 IP addresses expanded into /24 prefixes. Such a master blacklist could be useful for an emergency response to a novel or large-scale attack. Overall, BLAG has higher recall than any single blacklist or their naïve aggregation and has 95–99% specificity. BLAG also outperforms PRESTA+L, a competing approach, by having higher specificity for the same recall. We thus believe that BLAG could significantly improve network security, and produce higher-quality blacklists than existing approaches.

Acknowledgment

This project is the result of funding provided by the Science and Technology Directorate of the United States Department of Homeland Security under contract number D15PC00184. 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 or the US Government. Minlan Yu is partially supported by CNS 1413978 by the National Science Foundation. The authors are grateful to anonymous reviewers for their helpful comments. Additional thanks to Omid Alipourfard, Krutika Jain, Sulagana Mukherjee, Aqib Nisar and Yu-Chuan Yen for their inputs in the earlier draft of the paper.

References

[1] Krebsonsecurity hit with record ddos. https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/, Sept 2016.

[2] Netlab: Mirai scanner feed. http://data.netlab.360.com/mirai-scanner/, Sept 2016.

[3] Alienvault Reputation System. https://www.alienvault.com/, Sept 2019.

[4] Automated ip reputation system for spammers. http://www.chaosreigns.com/iprep/, Sept 2019.

[5] Badips.com, an ip based abuse tracker. https://www.badips.com/, Sept 2019.

[6] Bambenek consulting feeds. http://osint.bambenekconsulting.com/feeds/, Sept 2019.

[7] Binary defense systems. https://www.binarydefense.com/, Sept 2019.

[8] Blocklist.de fail2ban reporting service. https://www.blocklist.de/en/index.html, Sept 2019.

[9] Botscout we catch bots so that you don't have to. https://www.botscout.com, Sept 2019.

[10] Charles b. haley. http://charles.the-haleys.org/, Sept 2019.

[11] Cinsscore. http://ciarmy.com/, Sept 2019.

[12] Cisco talos. http://www.talosintelligence.com/, Sept 2019.

[13] Clean mx - realtime db. https://web.archive.org/web/20161102031447/http://clean-mx.com:80/, Sept 2019.

[14] Cloud spam protection for forums, boards, blogs and sites. https://www.cleantalk.org, Sept 2019.

[15] Criticalstack - the secure container orchestration platform for the enterprise. https://criticalstack.com/, Sept 2019.

[16] Cruzit - server blocklist / blacklist. http://www.cruzit.com/wbl.php, Sept 2019.

[17] Cybercrime. http://cybercrime-tracker.net/, Sept 2019.

[18] Daniel gerzo, bruteforceblocker. http://danger.rulez.sk/index.php/bruteforceblocker/, Sept 2019.

[19] Darklist.de | a malicious ip catcher blacklist. http://darklist.de/, Sept 2019.

[20] Dyn malware feeds. http://security-research.dyndns.org/pub/malware-feeds/, Sept 2019.

[21] Emerging Threats. https://rules.emergingthreats.net/, Sept 2019.

[22] General - blocklist.net.ua. https://blocklist.net.ua/, Sept 2019.

[23] Graphiclineweb. https://graphiclineweb.wordpress.com/tech-notes/ip-blacklist/, Sept 2019.

[24] Greensnow. https://greensnow.co/, Sept 2019.

[25] hphosts - by malware bytes. https://hosts-file.net/, Sept 2019.

[26] I-blocklist. https://www.iblocklist.com/, Sept 2019.

[27] Improware. http://antispam.imp.ch/, Sept 2019.

[28] Lashback blacklist. http://blacklist.lashback.com/, Sept 2019.

[29] Mailinator inc. http://www.mailinator.com, Sept 2019.

[30] Malc0de Database. http://malc0de.com/database/, Sept 2019.

[31] Malware domain list. http://www.malwaredomainlist.com/, Sept 2019.

[32] Maxmind - sample list of high risk ip addresses. https://www.maxmind.com/en/high-risk-ip-sample-list, Sept 2019.

[33] My ip - blacklist checks. https://www.myip.ms/info/about, Sept 2019.

[34] Nothink individual blacklist maintainer. http://www.nothink.org/, Sept 2019.

[35] Novirusthanks: Security software and services. http://www.ipspamlist.com/ip-reputation-feeds/, Sept 2019.

[36] OpenBL - Abuse reporting and blacklisting. https://web.archive.org/web/20170107081656/http://www.openbl.org/, Sept 2019.

[37] Project Honeypot. https://www.projecthoneypot.org/, Sept 2019.

[38] Project turris. https://www.turris.cz/en/greylist, Sept 2019.

[39] Proofpoint inc. http://www.proofpoint.com, Sept 2019.

[40] Sblam! http://sblam.com/, Sept 2019.

[41] Shun list. https://www.autoshun.org/, Sept 2019.

[42] Sourcefire vrt labs. https://labs.snort.org/, Sept 2019.

[43] Spamassasin public corpus. https://spamassassin.apache.org/old/publiccorpus/, Sept 2019.

[44] Spamhaus Project. https://www.spamhaus.org/, Sept 2019.

[45] Stefan gofferje - sip attacker blacklist. http://stefan.gofferje.net/it-stuff/sipfraud/sip-attacker-blacklist, Sept 2019.

[46] Stop forum spam. https://stopforumspam.com/, Sept 2019.

[47] Swiss Security Blog - Abuse.ch. https://www.abuse.ch/, Sept 2019.

[48] Trustedsec - information security consulting services. https://www.trustedsec.com/, Sept 2019.

[49] Untroubled. http://untroubled.org/spam/, Sept 2019.

[50] Urlvir: Monitor malicious executable urls. http://www.urlvir.com/, Sept 2019.

[51] Voip blacklist. http://www.voipbl.org/, Sept 2019.

[52] Vx vault. http://vxvault.net/ViriList.php, Sept 2019.

[53] 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.

[54] Christoph Dietzel, Anja Feldmann, and Thomas King. Blackholing at ixps: On the effectiveness of ddos mitigation in the wild. In *International Conference on Passive and Active Network Measurement*, pages 319–332. Springer, 2016.

[55] 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.

[56] Anushah Hossain, Sivaramakrishnan Ramanthan, and Sadia Afroz. Poster: Perceptions of third part blacklists. https://www.dropbox.com/s/i37ze8g4804owwz/usenix_poster.pdf?dl=0, August 2019.

[57] SANS Institute. Internet Storm Center. https://dshield.org/about.html, Sept 2019.

[58] ANT ISI. https://nsnam.isi.edu/datasets/readmes/b_root_anomaly-20160625.readme.txt. https://nsnam.isi.edu/datasets/readmes/B_Root_Anomaly-20160625.README.txt, June 2019.

[59] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[60] 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.

[61] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A research-oriented top sites ranking hardened against manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, NDSS 2019, 2019.

[62] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[63] Vector Guo Li, Matthew Dunn, Paul Pearce, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. Reading the tea leaves: A comparative analysis of threat intelligence. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 851–867, Santa Clara, CA, August 2019. USENIX Association.

[64] Justin Mason. Filtering spam with spamassassin. In *HEANet Annual Conference*, page 103, 2002.

[65] David Moore, Colleen Shannon, et al. Code-red: a case study on the spread and victims of an internet worm. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 273–284. ACM, 2002.

[66] Arman Noroozian, Jan Koenders, Eelco van Veldhuizen, Carlos H. Ganan, Sumayah Alrwais, Damon McCoy, and Michel van Eeten. Platforms in everything: Analyzing ground-truth data on the anatomy and economics of bullet-proof hosting. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 1341–1356, Santa Clara, CA, August 2019. USENIX Association.

[67] Krebs on Security. memcached attack. https://krebsonsecurity.com/tag/memcached-attack/, March 2018.

[68] Heise Online. NiX Spam DNSBL and blacklist for download. https://www.heise.de/ix/NiX-Spam-DNSBL-and-blacklist-for-download-499637.html, Sept 2019.

[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] Oregon RouteViews. University of oregon routeviews project. *Eugene, OR.[Online]. Available: http://www. routeviews. org*.

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

[73] Quirin Scheitle, Oliver Hohlfeld, Julien Gamba, Jonas Jelten, Torsten Zimmermann, Stephen D Strowes, and Narseo Vallina-Rodriguez. A long way to the top: significance, structure, and stability of internet top lists. In *Proceedings of the Internet Measurement Conference 2018*, pages 478–493. ACM, 2018.

[74] 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.

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

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

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

[78] 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.

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

[80] 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.

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