Internet Bad Neighborhoods Aggregation

Giovane C. M. Moura, Ramin Sadre, Anna Sperotto, and Aiko Pras
University of Twente
Centre for Telematics and Information Technology (CTIT)
Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
Design and Analysis of Communications Systems (DACS)
Enschede, The Netherlands
Email: {g.c.m.moura, r.sadre, a.sperotto, a.pras}@utwente.nl

***Abstract*—Internet Bad Neighborhoods have proven to be an innovative approach for fighting spam. They have also helped to understand how spammers are distributed on the Internet. In our previous works, the size of each bad neighborhood was fixed to a /24 subnetwork. In this paper, however, we investigate if it is feasible to aggregate Internet bad neighborhoods not only at /24, but to any network prefix. To do that, we propose two different aggregation strategies: fixed prefix and variable prefix. The motivation for doing that is to reduce the number of entries in the bad neighborhood list, thus reducing memory storage requirements for intrusion detection solutions. We also introduce two error measures that allow to quantify how much error was incurred by the aggregation process. An evaluation of both strategies was conducted by analyzing real world data in our aggregation prototype.**

I. Introduction

Malicious hosts are not evenly distributed on the Internet, but they tend to be concentrated in certain subnetworks [1]–[3]. The rationale behind this observed behavior is that different subnetworks have different security policies, and poorly managed subnetworks tend to let their hosts more vulnerable than hosts belonging to well managed networks. In [4], van Wanrooij and Pras investigated how to exploit this behavior to enhance spam filters and introduced the *Internet Bad Neighborhood* concept. An Internet Bad Neighborhood (BadHood in the rest of this work) is a netblock[1] of a certain size to which a certain number of misbehaving hosts belong. The idea behind BadHoods is that the probability that a specific IP address behaves badly increases if neighbor IP addresses, *i.e.*, hosts within the same netblock, behave badly.

In a previous research work [5], we have addressed Spamming Bad Neighborhoods, identifying emerging behaviors from the analysis of spamming netblocks. Badhoods have proven to be a meaningful approach to characterize the behavior of a large portion of the address space, and one of their possible application is in the creation of *blacklists*, i.e., lists of IP addresses (or, in our case, netblocks) that are likely to be involved in malicious activities. The same reasoning can be extended to other types of malicious attacks/applications, such as *phishing*, Distributed Denial of Service (DDoS), and SSH brute force attacks.

[1]By "netblock" we mean here a set of IP addresses sharing the same prefix.

978-1-4673-0269-2/12/$31.00 © 2012 IEEE

In both previous research works [4], [5], Spamming BadHoods were evaluated only under a single fixed block size, i.e., we considered BadHoods consisting of a single /24 block (254 /32 hosts). We have initially chosen to employ /24 because it is the "minimum prefix size which is generally routable on the Internet" [6]. Even though the /24 prefix has proven effective for spam filtering and to identify Spamming BadHood behaviors, there might be cases in which some BadHoods can be aggregated into larger netblock sizes.

The motivation for doing it is that using larger blocks would reduce the number of entries in a BadHood blacklist. This principle is analogous to the reduction of entries in routing tables when Classless Inter-Domain Routing (CIDR) [7] was introduced. If Internet BadHoods-based defense mechanisms are supposed to operate efficiently in real-time, the aggregation of entries would reduce memory storage requirements and the look-up time on the BadHood blacklists. To give an idea of the problem, current /24 Spamming BadHoods may have more than one million entries, while Border Gateway Protocol (BGP) routing tables [8] have no more than 385k entries [9].

To better illustrate the concept of aggregation of Internet BadHoods, take as example the crime rate reports of the Police Department of a city such as New York [10]. Consider that the Police department has security policies for neighborhoods (or "precinct", term used by the police) that vary according to their respective crime rates. Also, consider two neighboring precincts: the 26th and 28th, from the Manhattan North, having different security policies. If these two were, then, found having similar crime rates, would it not be useful to consider them as a single "bigger" precint and employ the same policy for both, thus simplifying the management? The same reasoning applies to Internet BadHoods. BadHoods exhibiting similar behavior can be aggregated and managed in the same way.

In this paper, we investigate (*i*) how to meaningfully aggregate BadHoods with prefixes other than /24, and (*ii*) what is the effect of the aggregation on the precision of the resulting BadHoods. We propose and evaluate two approaches to BadHoods aggregation. Firstly, we assess the impact of a *fixed prefix aggregation strategy*, which produces blacklists in which all the BadHoods have the same prefix, and consequently the same size. After that, we evaluate the *variable prefix aggregation strategy*, in which BadHoods of different sizes coexist in the same blacklist. Real world security data is employed in the evaluation of both strategies, which is done by showing both the gain in blacklist size reduction, as well as the precision of the resulting blacklists.

The organization of this paper is as follows. In Section II we introduce the metrics that are employed to evaluate each BadHood during the aggregation process. Next, in Section III, we present the strategies for aggregating Internet BadHoods. Following that, in Section IV we evaluate those strategies by using real world data. The related work is then presented on Section V, and the summary and conclusions are shown in Section VI.

II. BadHoods Evilness Metrics

Typical data for defining Bad Neighborhoods are lists of IP addresses which have been flagged as sources of malicious activities [5]. They can be obtained from defense and monitoring mechanisms, such as intrusion detection systems (IDSs) [11], [12] or honeypots. In some cases, third parties provide blacklists containing IP addresses of malicious hosts. A major example in this case are DNS Blacklists [13], built by harvesting spamming IP addresses using *spamtraps* (specialized honeypots to collect spam) distributed over different domains. Examples include SBL [3], CBL [14], and PSBL [15]. The same applies to Phishing URL data sources, such as the list provided by PhishTank [16] and for SSH brute-force blacklists, such as SSHBL.org [17].

Given a list of malicious IP addresses (/32), we define a $/n$ BadHood (in CIDR prefix notation [7]) as a $/n$ netblock A^n with a score $score(A^n)$, where the score is the number of malicious hosts in the block:

$$score(A^n) = \#\{\text{malicious hosts in block } A^n\} \tag{1}$$

In [5], we created /24 BadHoods employing different spammers data sources. As in the previous work, in this one we also begin with this aggregation level since a /24 is the minimum prefix "routable on the Internet" [6]. Table I provides a short example of /24 BadHoods.

#	**/24 netblock**	**Score**
1	10.10.10.0	22
2	10.10.11.0	21
3	10.10.12.0	20
4	10.10.13.0	41
5	20.20.24.0	130
6	20.20.25.0	1
7	30.30.34.0	60

TABLE I
Example of /24 BadHoods and their scores

The score value leads to an intuitive definition of the "evilness" of a netblock: the higher the score, the higher the probability that a single host address (/32) from the /24 block is a source of malicious activities. However, the score depends on the size of the block. Therefore, it is useful to have a normalized measure of evilness regardless the block size. Let A^n be a netblock of size $/n$ with score $score(A^n)$. We define the *infection rate* of A^n as

$$p_n(A^n) = \frac{score(A^n)}{max\_hosts(A^n)}, \tag{2}$$

where $max\_hosts(A^n) = 2^{32-n}$ is the maximum number of IP addresses in a $/n$ netblock (neglecting the addresses reserved for broadcast and network identification).

In the next section we present the aggregation strategies and how these metrics are employed in the aggregation process.

III. Internet BadHoods Aggregation Strategies

In this section, we describe how /24 BadHoods can be aggregated into larger BadHoods blocks (or smaller prefixes). We begin by introducing the aggregation operation in Section III-A, which formalizes the process of merging two $/n$ BadHoods into one $(/n-1)$ BadHood.

Based on this operation, we then present two iterative strategies to aggregate a complete list of /24 BadHoods. The first strategy increases in each iteration the size of the BadHoods by a factor of two by merging adjacent BadHoods. We refer to this aggregation strategy as *fixed prefix aggregation* and present it in Section III-B. This strategy can considerably reduce the number of BadHoods but also decreases the precision of the resulting BadHood list. To overcome this issue, we investigate a *variable prefix aggregation* strategy in Section III-C. In contrast to the first strategy, this strategy merges adjacent BadHoods *only if* they fulfill the proposed merging condition.

A. The Basic Aggregation Operation

Given two $/n$ BadHoods A^n and B^n can be aggregated into the $/(n-1)$ BadHood $A^n \oplus B^n$ if A^n and B^n have a common address prefix of $n-1$ bits. The aggregated BadHood $A^n \oplus B^n$ spans the IP addresses of A^n and B^n. For example, in Table I, blocks #1 and #2 can be aggregated from /24 to /23, while blocks #1 and #7 can not.

Consequently, the infection rate of the aggregated BadHood $A^n \oplus B^n$ is as follows:

$$p_{n-1}(A^n \oplus B^n) = \frac{score(A^n) + score(B^n)}{max\_hosts(A^n \oplus B^n)} = \frac{1}{2}\left(p_n(A) + p_n(B)\right). \tag{3}$$

B. Fixed Prefix Aggregation Strategy

The *fixed prefix aggregation* strategy iteratively aggregates BadHoods to larger netblocks. In the first iteration, all /24 BadHoods are aggregated to /23 BadHoods according to the aggregation operation defined in the previous subsection. For example, the /24 BadHoods provided in Table I will be aggregated into the /23 BadHoods shown in Table II. In the next iteration the /23 BadHoods are aggregated to /22 ones, and so on.

It is important to note that, in this strategy, we *always* aggregate BadHoods: BadHoods that have no "partner" BadHood to be aggregated with are aggregated with an empty netblock, i.e., with a netblock of score 0 (as defined in Equation (1)).

#	/23 netblock	score
1	10.10.10.0/23	43
2	10.10.12.0/23	61
3	20.20.24.0/23	131
4	30.30.34.0/23	60

TABLE II
/23 BADHOODS RESULTING FROM FIXED PREFIX AGGREGATION

In our example, the /24 BadHood 30.30.34.0 in Table I is aggregated with the zero-score netblock 30.30.35.0.

Algorithm 1 presents the pseudocode for this strategy. The algorithm takes as input the initial list S_{24} of /24 netblocks B_i^{24} with $score(B_i^{24})$ and the largest desired aggregation level m. In each iteration (line 1), the algorithm builds the list S_{n-1} of $/(n-1)$ BadHoods by merging all pairs of $/n$ BadHoods B_i^n, B_j^n according to the basic aggregation operation given in Section III-A (lines 3–5). Empty netblocks are included if no matching BadHood is found for aggregation (line 3).

Algorithm 1 Fixed prefix aggregation

```
Input: $S_{24} = \{(B_i^{24}, score(B_i^{24})), i = 1 \ldots num\_entries\}$
Input: largest aggregation level $m$
Output: $S_m$
1: for $n = 24 \rightarrow m+1$ do
2:     $S_{n-1} := \emptyset$
3:     for all $B_i^n, B_j^n \in S_n, i \neq j$ with common $n-1$ prefix (use
       an empty netblock if no matching $B_j^n$ found) do
4:         $S_{n-1} := S_{n-1} \cup \{(B_i^n \oplus B_j^n, score(B_i^n \oplus B_j^n))\}$
5:     end for
6: end for
```

The fixed prefix aggregation strategy effectively reduces the number of BadHoods in each iteration because it progressively builds larger netblocks regardless their scores. However, this simple approach also exhibits some drawbacks. First, aggregating two BadHoods with infection rates a and b will result in a BadHood with an infection rate of $\frac{a+b}{2}$. The larger the difference between a and b, the more information about the behavior of the individual /24 BadHoods in the aggregated BadHood is lost. Secondly, enlarging the BadHoods can have the side effect of including also netblocks that were not initially flagged as malicious, as already illustrated in our example by the netblock 30.30.35.0. This effect aggravates with each iteration. This

C. Variable Prefix Aggregation Strategy

Differently from the fixed prefix aggregation strategy, our next strategy does not apply the same degree of aggregation to all BadHoods. Instead, the main idea is to merge two BadHoods only if they satisfy a *merging condition*. Intuitively, the merging condition should ensure that the BadHoods to be merged are sufficiently similar and, hence, the aggregated BadHood is, to some extend, representative for them.

Algorithm 2 presents the pseudocode for the proposed aggregation strategy. As in the previous strategy, the algorithm takes as input the initial list S_{24} of /24 netblocks B_i^{24} with $score(B_i^{24})$ and the largest desired aggregation level m. Then, for each aggregation level n (line 2), the algorithm merges all $/n$ BadHoods B_i^n, B_j^n which would form a valid aggregated BadHood according to the basic aggregation operation (see Section III-A) that satisfy the merging condition (line 3). BadHoods that do not fulfill those conditions are not aggregated and therefore not considered further for aggregation in this or the next iterations.

Algorithm 2 Variable prefix aggregation

```
Input: $S_{24} = \{(B_i^{24}, score(B_i^{24})), i = 1 \ldots num\_entries\}$
Input: largest aggregation level $m$
Input: merging condition parameter $\beta$
Output: $S$
1: $S := S_{24}$
2: for $n = 24 \rightarrow m+1$ do
3:     for all $B_i^n, B_j^n \in S, i \neq j$ with common $n-1$ prefix
       $\wedge\ merge(A^n, B^n)$ do
4:         $S := S \setminus \{(B_i^n, score(B_i^n)), (B_j^n, score(B_j^n))\} \cup \{(B_i^n \oplus$
           $B_j^n, score(B_i^n \oplus B_j^n))\}$
5:     end for
6: end for
```

The merging condition is defined as

$$merge(A^n, B^n) = p_{n-1}(A^n \oplus B^n) \geq \beta \cdot \max(p_n(A), p_n(B)). \quad (4)$$

The condition is such that we allow a merge only if the resulting infection rate $p_{n-1}(A^n \oplus B^n)$ is at least equal to a fraction β of the most malicious of the blocks to be merged. The parameter β prevents therefore the aggregation strategy from merging dissimilar BadHoods. This value can be tuned according to the scenario and application. β ranges between 0.5 and 1.0: smaller values make the aggregation less strict, thus allowing more BadHoods to be merged. Values close to 1 will instead lead to a less permissive aggregation strategy.

Finally, at line 4, the algorithm progressively builds the new BadHood set by removing BadHoods and replacing them with the merged one. Note that, in contrast to the fixed prefix aggregation strategy, now BadHoods of different sizes coexist in the result set S.

In order to illustrate the strategy, we apply it to the example given in Table I. For $\beta = 0.8$, we obtain after one iteration the BadHoods shown in Table III. Blocks #1 and #2 are merged, because $p_{24}(\#1) = \frac{22}{254}$, $p_{24}(\#2) = \frac{21}{254}$, and $p_{23}(\#1 + \#2) = \frac{43}{510}$, so $p(\#1 + \#2) > 0.8 \cdot max(\cdot) \Rightarrow 0.086 > 0.069$. The other blocks, on the other hand, do not match the condition, so they are not aggregated. After the first iteration, the list contains both /23 and /24 entries. In the next iterations, no further aggregation occurs, and the final result contains entries using mixed prefixes (/23 and /24).

Comparing the results of both strategies (Tables II and III), the output of the variable prefix strategy has more entries. However, the blocks aggregated by the variable prefix strategy have been matched against a stricter merging criteria. In the next section we evaluate both strategies using real world data.

#	/23 netblock	score
1	10.10.10.0/23	43
2	10.10.12.0/24	20
3	10.10.13.0/24	41
4	20.20.24.0/24	130
5	20.20.25.0/24	1
6	30.30.34.0/24	60

TABLE III
BADHOODS RESULTING FROM VARIABLE PREFIX AGGREGATION

Finally, it should be noted that the above algorithms can be efficiently implemented without employing set operations. In our Java prototype, we have observed runtimes of less than 10 seconds even for very large input files ($>$ 1M entries).

IV. AGGREGATION STRATEGIES EVALUATION

The main benefit of aggregating Internet BadHoods is to reduce the number of lines in the original BadHood blacklist. However, this reduction comes with a price: hosts belonging to aggregated netblocks might be seen as more or less malicious than what was observed before the aggregation. In this section we evaluate and analyze the impact of the aggregation strategies.

We firstly present the data sets that we use for our evaluation in Section IV-A. Then, we introduce the performance metrics that we use to evaluate the strategies in Section IV-B. The performance of both strategies is compared for the largest of our datasets, the Composite Blocking List (CBL; see below), in Section IV-C. In Section IV-D, we study the impact of the merging parameter β on the performance of the variable prefix aggregation strategy. Finally, we compare the results for different datasets in Section IV-E.

A. BadHood Input Blacklists

We evaluate our aggregation strategies on the real case of a Spam blacklist. The considered data set is the Composite Blocking List (CBL) [14] – an online Spam DNS blacklist. CBL maintains four large spamtrap infrastructures from where the source IP addresses of spammers are harvested. We have obtained the list for the April 28th, 2010. On this day, CBL listed 8,177,138 /32 IP addresses, which resulted in an initial blacklist of 960,167 /24 BadHoods, as explained in Section II.

In addition to the above list, we use the following datasets for our experiments in Section IV-E:

- Passive Spam Block List (PSBL) (2010) [15], obtained on April 28th, 2010: the list consists of more than 2.8M /32 distinct IP addresses;
- Passive Spam Block List (PSBL) (2011) [15], obtained on October 24th, 2011: the list consists of more than 283K /32 distinct IP addresses;
- Mail server logs from Provider A: Provider A is a major hosting provider in the Netherlands. We have obtained the IP addresses of spammers on April 28th, 2010. For this day, 256K distinct /32 IP addresses were observed.

B. BadHood Aggregation Performance Metrics

Our main aggregation performance metric of interest is clearly the reduction achieved by the strategies in terms of number of entries in the initial BadHood input list. We measure it as the difference between the number of entries (also called "lines" in the following) in the initial /24 BadHood list and in the resulting list generated by the algorithms in Section III.

As already mentioned previously, the achieved reduction comes along with a loss of information on the behavior of the individual /24 BadHoods. Consider as an example an application, such as a Spam filter, that relies on the aggregated lists. Let be $\{X^{24}, Y^{24}, \ldots\}$ a set of /24 BadHoods with infection rates $\{p_{24}(X^{24}), p_{24}(Y^{24}), \ldots\}$. We can interpret $p_{24}(X^{24})$ as the probability that a particular IP address in block X^{24} be a source of malicious activities. After we have aggregated the /24 BadHoods to a $/n$ BadHood A^n, with $n < 24$, only the infection rate $p_n(A^n)$ of the aggregated BadHood is available to the application. The "evilness" of a particular IP address in X^n can now only be estimated by $p_n(A^n)$ (Equation (2)). Consequently, we define the error $err(X^{24})$ introduced by the aggregation for the BadHood X^{24} as

$$err(X^{24}) = p_n(A^n) - p_{24}(X^{24}). \tag{5}$$

A positive (negative) error indicates that the application would overestimate (underestimate) the evilness of X^{24} after the aggregation. To assess the global error for a whole blacklist, we sum up the absolute errors for each /24 BadHood X^{24}:

$$Err_{abs} = \sum |err(X^{24})| \tag{6}$$

Alternatively, we sum the squares of the individual errors:

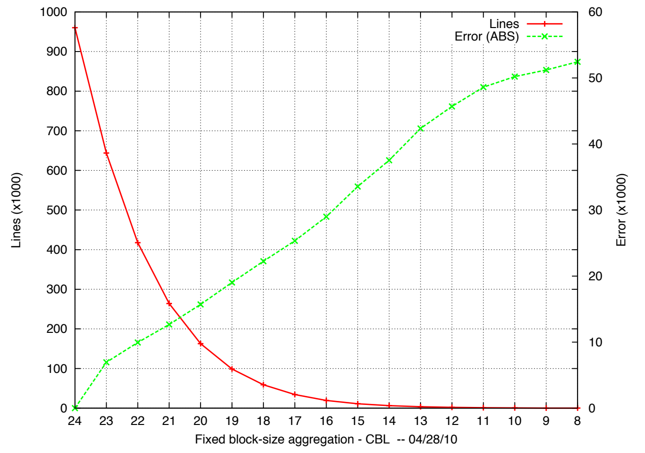
$$Err_{square} = \sum err(X^{24})^2 \tag{7}$$

To illustrate how they are calculated, consider block #1 in Table II. Before being aggregated into a /23 BadHood, the infection rate $p(\#1)$ was $\frac{22}{256}$ (Table I). After that, the same netblock gets the mean value $\frac{43}{512}$. The error $err(\#1)$ introduced is therefore, $\frac{22}{256} - \frac{43}{512} = -0.0019$. After calculating the individual errors, the global errors, as defined in Eq. (6) and (7), can be obtained.

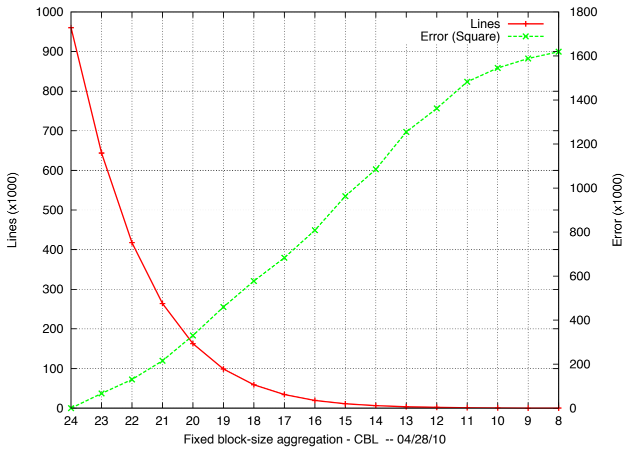
The interpretation of the error values depends on the application and other definitions of the global errors are possible. For example, for an intrusion detection system, large error measures may cause increased False Positive or False Negative rates. In such scenario, calculating the global errors separately for positive and negative $err(X^{24})$ values could be of interest. In order to be independent from a particular application and suitable for different scenarios, we have chosen the rather flexible definitions in Eq. (6) and (7).

C. Performance of the Aggregation Strategies

In this section, we present the results of the aggregation strategies applied to the CBL data set. We first analyze the reduction on the size of the blacklist as result of the aggregation. Then, we discuss the impact of the aggregation strategies on the global errors, as defined in Section IV-B.

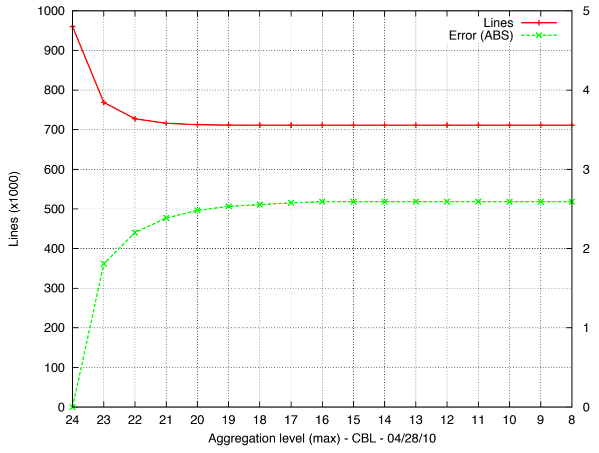


(a) Absolute error

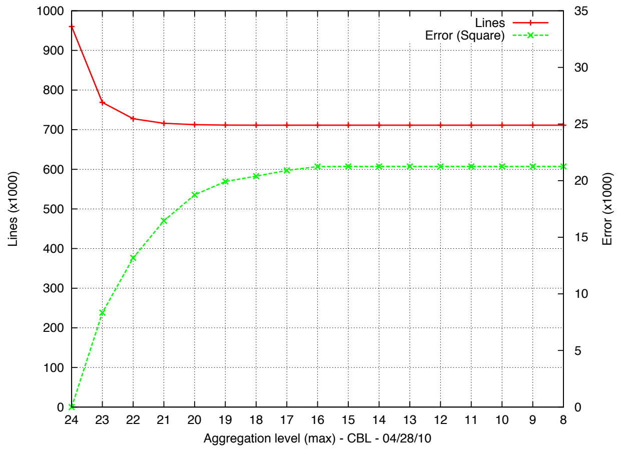


(b) Square error

Fig. 1. Fixed prefix aggregation strategy



(a) Absolute error for β=0.8



(b) Square error for β=0.8

Fig. 2. Variable prefix aggregation strategy

1) Blacklist size: Figures 1(a) shows the number of entries (in thousands) in the result blacklists as function of the aggregation level *m* for the fixed prefix aggregation strategy, while Figure 2(a) shows it for the variable prefix aggregation strategy. For the later, we have chosen a rather moderate merging parameter of $\beta = 0.8$. The influence of the parameter is discussed in Section IV-D.

As expected, both strategies are able to reduce the number of entries of the initial input blacklist. If compared, however, we can see that their performance is very dissimilar. The fixed prefix strategy progressively aggregates listed BadHoods into larger netblocks, regardless of their scores and infection rates. As a result the number of entries decreases with the aggregation level, i.e., with increasing block sizes.

The variable prefix strategy, on the other hand, only aggregates blocks that meet the merging condition specified in Eq. (4). As a result, once no more candidate blocks satisfy the condition, the number of entries in the blacklist stabilizes. As can be seen in Figure 2(a) we observe that there is no more aggregation after /15 prefix. Indeed, most of the aggregation is achieved when moving from level /24 to /23. Hence, the variable prefix aggregation can be seen as a "less aggressive" approach than the fixed prefix one. While the fixed prefix strategy reduces the blacklist, from the original 960,167 /24 BadHoods to 162 entries for /8, the variable prefix strategy stabilizes at around 711K entries.

2) Error: Figures 1(a) and 2(a) also show the global absolute errors (see Eq. (6)) for the two strategies as function of the aggregation level. The results for the global square errors (see Eq. (7)) are shown in Figures 1(b) and 2(b).

First and foremost, we observe that the fixed prefix aggregation strategy results in much larger errors than the variable prefix aggregation strategy. This is an expected result since the former aggregates blocks regardless of their infection rate. Therefore, many dissimilar blocks (in regards to their scores in Eq. (1)) are aggregated, leading to large differences between the infection rates of the individual /24 blocks and the infection rate of the aggregated block.

In the case of the fixed prefix aggregation strategy, we also observe that the errors almost linearly increase with the (decreasing) aggregation level, although the achieved reduction of the number of lines is not linear at all. This is due to the fact that the strategy also considers empty blocks, i.e., blocks with a score of 0. Aggregating an empty block with a non-empty block does not reduce the number of lines, but increases the error. This effect aggravates with the aggregation level (see Section III-B). In fact, up to around aggregation level /18, substantial reductions in the number of lines are achieved by the strategy. After this point, as we can see from the constant error increase, the aggregation of two netblocks becomes more expensive in terms of errors. At aggregation level /8, the absolute and square errors are respectively 2.36 and 2.8 time larger than at the /18 level. This leads to the conclusion that the aggregation to larger netblocks (small prefixes) has a huge impact on the precision of the final blacklist and therefore it might not be worthy to aggregate beyond a certain level.

In contrast, the error curves of the variable prefix aggregation strategy mostly mirror the achieved reduction of lines. Both the number of lines and the global errors significantly change up to around level /18. After this point, there are nearly no more aggregations and the error stabilizes. The square error (see Figure 2(b)) better visualizes the changes at the levels /21 through /17 than the absolute error (see Figure 2(a)).

3) Distribution of Malicious Hosts: As already stated, the variable prefix aggregation strategy achieves most of the reduction when moving from level /24 to /23. After that, only a small portion of the BadHoods fulfill the merging condition and can be aggregated further. The bar chart in Figure 3 (left y-axis) shows the resulting distribution of the BadHood sizes for aggregation level $m = 8$ and $\beta = 0.8$. As can be seen, a large portion (around 580k) of the initial 960,167 /24 BadHoods are not aggregated at all and remain at level /24. Around 109k entries are aggregated to /23 BadHoods and only a few entries are aggregated to /22 or higher.

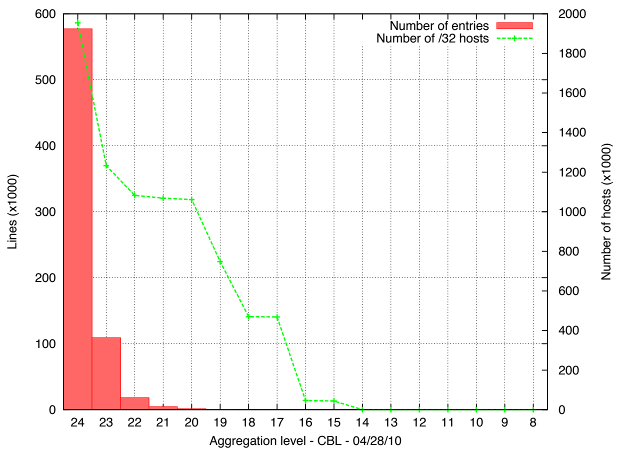


Fig. 3. Distribution of BadHoods and host addresses for aggregation level /8 for $\beta = 0.8$

In the same figure, we also show the distribution of the number of /32 host addresses (right y-axis). Remember, that the original data set contains 8,177,138 host addresses (see Section IV-A). According to the figure, around 2 million host addresses remain in /24 BadHoods after aggregation, but most of the hosts can now be found in /23 to /17 BadHoods. Surprisingly, the distribution of the host addresses over the aggregated BadHoods does not match the distribution of the BadHood sizes, even when considering that a $/(n-1)$ BadHood is twice as large as a $/n$ one. For example, the /23 BadHoods has around 60% more host addresses than what one would expect from the BadHood size distribution. This can be explained by the nature of BadHoods. Since a /24 netblock with a high score indicates a badly managed subnetwork, it is natural to assume that similar netblocks can be found in its neighborhood. Such netblocks are preferred by the merging condition in Eq. (4) and, hence, are more likely aggregated. These results illustrate the benefits of the aggregation.

D. The Impact of the Parameter β on the Aggregation

In the following experiments we study the impact of the merging parameter β on the performance of the variable prefix aggregation strategy. Intuitively, if β is too permissive (β close to 0.5), it might results in a blacklist in which most of the blocks are aggregated, while a more strict value for β (β close to 1.0) would aggregate very few blocks. However, at the same time, a permissive aggregation strategy would also result in a larger aggregation error, while a strategy aggregating only very similar blocks would result in a small error.

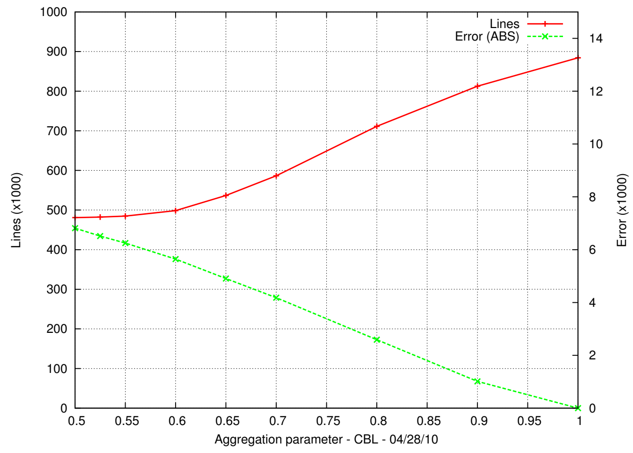
Figure 4 shows, for varying values of β, the number of entries in the blacklist output by the aggregation strategy, as well as the corresponding global errors (absolute and square). For $\beta = 0.5$, the resulting BadHood blacklist contains around 470k entries. For increasing values of β, the blacklist becomes progressively larger, while the errors decrease almost linearly. Finally, for $\beta = 1.0$, the final blacklist has almost the same number of lines as the original not aggregated one, since the algorithm only aggregates valid netblocks with exactly the same infection rate. Therefore, no error is observed for $\beta = 1.0$.

The figure also clearly shows that there is a trade-off between having a short and efficient blacklist and having a small merging error. Therefore an appropriate value of β should be chosen case by case, and, accordingly to the scenario, the security manager should decide to compromise incurred error and blacklist size.

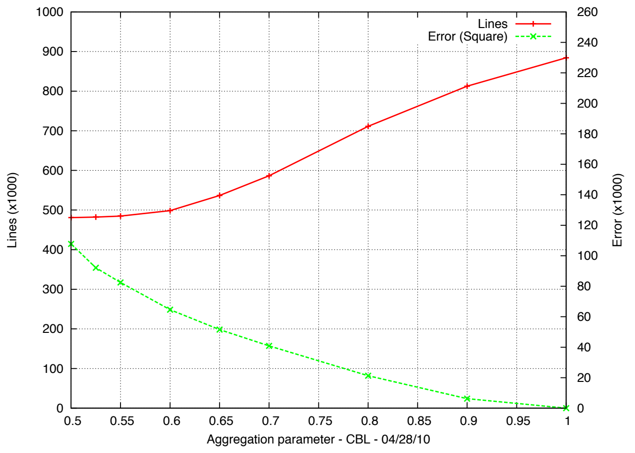
E. Aggregating Different Input Blacklists

We now discuss the performance of the variable prefix aggregation strategy for the data sets presented in Section IV-A. We omit the result of the fixed prefix aggregation strategy due to its large errors. In general, the same behavior as in Section IV-C is observed.

In Figure 5, we show the number of lines of the result blacklists relative to the original sizes of the /24 data sets, as computed by the variable prefix aggregation for varying aggregation level and $\beta = 0.8$. We observe that our aggregation



(a) ABS Error



(b) Square Error

Fig. 4. The impact of β on the variable prefix aggregation

strategy is able to reduce the blacklist size for each of the considered data sets. For $\beta = 0.8$, the data sources experience a reduction on the number of entries from 10% for the "Provider A" data set to 26% for the CBL.

A second observation is that the two largest lists, namely CBL and PSBL 2010 (April 28th), clearly benefit more from the aggregation than the smaller lists. This is expected because the BadHoods in the smaller lists are more sparsely distributed over the Internet address space and, hence, are harder to aggregate. In addition, the "Provider A" data set experiences the smallest reduction of all four traces.

V. Related Work

To the best of our knowledge, this is the first work that addresses Internet Bad Neighborhoods and IP aggregation in combination. However, other research works have addressed both topics separately. Therefore, in this section we first cover works related to Internet Bad Neigborhoods and then those

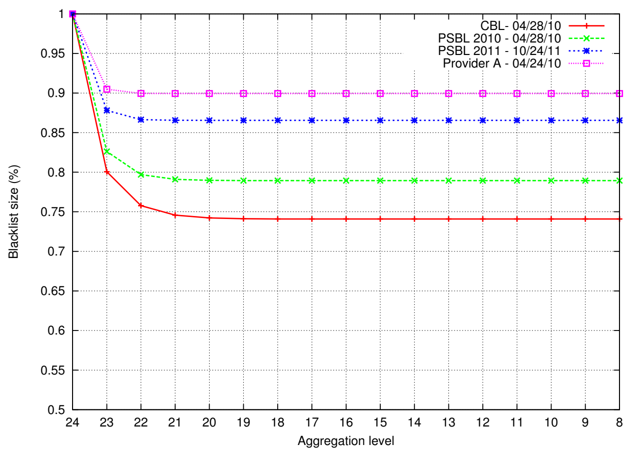


Fig. 5. Variable prefix aggregation strategy applied to different data sets (β=0.8)

that address IP aggregation.

Previous studies [1], [2] have shown that malicious hosts tend to be concentrated on some subnetworks instead of being randomly distributed on the Internet. DNS blacklists [13], such as [3], [14], [15], also suggest the same concentration. Taking this into account, the Bad Neighborhood concept was introduced in [4]. In that study, the authors have developed a mail filter that employed Spam BadHoods to tell whether a particular message was spam or not, based on the sender's IP address.

In a following work, we have investigated in details Spamming BadHoods [5]. By taking into account the way spammers behave [18], we have proposed four definitions for Spamming BadHoods, each of them addressing a particular part of the "spam picture". Among the results, we have found that even though bots seems to be responsible for most of the spam, we should not neglect the impact by massive spamming BadHoods, that is, spamming BadHoods that have few hosts sending a massive number of spam each to a single domain. In the two previous research works Spamming BadHoods were evaluated using the /24 prefix. In this work, however, we present two strategies that allow to build Internet BadHoods using other prefixes than /24 that can be employed as well to analyze Spamming BadHoods and any other type of Internet Bad Neighborhood.

The other topic addressed in this paper – IP aggregation – was previously investigated by the IP routing community. Back in 1993, only classful addresses were used (former classes A, B, and C). This addressing scheme was causing several problems – including exhaustion of the Class B network address space and "growth of routing tables in Internet routers beyond the ability of current software, hardware, and people to effectively manage" [7]. Therefore, in 1993 the IETF introduced the Classless Inter-domain Routing (CIDR) addressing [7], and the prefix notation used in this work. This new addressing scheme allowed blocks to be allocated under prefixes different than the ones specified by classes A, B, and C. That allowed route entries with the same prefix to be aggregated in what is called supernets [19]. By aggregating them, the number of entries in routing tables of BGP [8] routers was reduced, and that decreased the requirements for storing route information on routers and the overhead when matching routes. Current BGP routers have typically 372k entries in their routing tables [9], a small value compared to current /24 BadHoods blacklists.

Other research works have been conducted in related areas. Liu *et al* [20] have proposed an algorithm to minimize rules in a firewall. Differently from our work, their algorithm is based on packet classifiers (e.g, fields) and not on network prefixes.

VI. Summary and Conclusions

Internet BadHoods are built upon the idea that hosts in a compromised environment also have a certain likelihood to be compromised in the same way. In previous studies, Internet BadHoods were investigated using a /24 aggregation level. In this paper, however, we question this pre-established value, and investigate how it is possible to meaningfully aggregate Bad Neighborhood blacklists and what is the effect of this operation compared to the original /24 Bad Neighborhoods. The motivation for doing that is to reduce the number of entries in BadHood blacklists, therefore improving their efficiency. However, aggregation might impact the precision of the blacklisting process, for example if a largely benign network block increases its malicious score due to an aggregation with a neighboring block.

We propose two different strategies to aggregate Internet BadHoods to different prefixes, and measured the effectiveness of the aggregation, as well as the introduced error. The fixed prefix aggregation strategy has proven to be very efficient when it comes to reducing the number of lines in Badhood Blacklists. By aggregating the entries to /18, we observe a reduction of 93.85% on the original size. However, the error incurred by this strategy is high. For the evaluated data, the results have shown that the aggregation after /18 corresponds to a reduction of very few entries at the expense of a large error. The variable prefix aggregation strategy can be seen, with respect to the final number of entries, as less aggressive, since blocks are only aggregated if they match the merging condition. Therefore, we observe a smaller size reduction compared to the fixed prefix aggregation, but the overall error is also reduced. Moreover, due to the merging condition, the variable prefix aggregation strategy terminates after few iteration, when no more aggregation are possible.

From the presented results we conclude that Internet BadHood aggregation is a viable approach to the problem of reducing the size of a blacklist. However, when applying the fixed size aggregation strategy we should be aware that this is likely to result in a large number of benign netblocks being blacklisted. Therefore, this strategy should be used only in cases where the BadHood size is more important than the error introduced. On the other hand, the variable size aggregation strategy can be employed in cases where it is required to compromise between error and size of the final blacklist. The aggregation parameter β was introduced in this strategy to allow one to fine tune the trade off between blacklist size and error. We have shown that by having 0.8 for β, we could reduce the number of entries by 10% to 26%, depending on the data source. However, the best value for β remains an application- and management-dependent decision.

Acknowledgments: This research work has been partially supported by the EU UniverSelf Collaborative Project (#257513) and the SURFnet's GigaPort3 project for Next-Generation Networks.

References

[1] A. Ramachandran and N. Feamster, "Understanding the Network-level Behavior of Spammers," in *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '06. New York, NY, USA: ACM, 2006, pp. 291–302.

[2] M. P. Collins, T. J. Shimeall, S. Faber, J. Janies, R. Weaver, M. De Shon, and J. Kadane, "Using Uncleanliness to Predict Future Botnet Addresses," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, ser. IMC '07. New York, NY, USA: ACM, 2007, pp. 93–104.

[3] The Spamhaus Block List, 2011. [Online]. Available: http://www.spamhaus.org/sbl/

[4] W. van Wanrooij and A. Pras, "Filtering Spam from Bad Neighborhoods," *International Journal of Network Management*, vol. 20, no. 6, pp. 433–444, November 2010.

[5] G. C. M. Moura, R. Sadre, and A. Pras, "Internet Bad Neighborhoods: The Spam Case," in *7th International Conference on Network and Services Management (CNSM 2011)*, Paris, France, 2011.

[6] RIPE NCC, "PI Assignment Size ," July 2006. [Online]. Available: http://www.ripe.net/ripe/policies/proposals/2006-05

[7] V. Fuller and T. Li, "RFC 4632: Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan," August 2006.

[8] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271 (Draft Standard), Internet Engineering Task Force, Jan. 2006, updated by RFC 6286.

[9] T. Bates, "CIDR Report," August 2011. [Online]. Available: http://www.cidr-report.org/as2.0/#General_Status

[10] NYPD, "NYPD: Office of the Chief of Department - Crime Prevention – Crime Statistics," August 2011. [Online]. Available: http://www.nyc.gov/html/nypd/html/crime_prevention/crime_statistics.shtml

[11] Snort, "Snort: A free lightweight network intrusion detection system for UNIX and Windows," 2011. [Online]. Available: http://www.snort.org

[12] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An Overview of IP Flow-Based Intrusion Detection," *Communications Surveys Tutorials, IEEE*, vol. 12, no. 3, pp. 343 –356, 2010.

[13] J. Levine, "DNS Blacklists and Whitelists," RFC 5782 (Informational), Internet Engineering Task Force, Feb. 2010.

[14] The CBL, 2011. [Online]. Available: http://cbl.abuseat.org/

[15] Passive Spam Block List, 2011. [Online]. Available: http://psbl.surriel.com/

[16] PhishTank, "PhishTank: Join the Fight Against Phishing," 2011. [Online]. Available: http://www.phishtank.com

[17] SSHBL.org, "sshbl.org - (the SSH blacklist)," 2012. [Online]. Available: http://http://www.sshbl.org/

[18] A. Pathak, Y. C. Hu, and Z. M. Mao, "Peeking into Spammer Behavior from a Unique Vantage Point," in *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*. Berkeley, CA, USA: USENIX Association, 2008.

[19] V. Fuller, T. Li, J. Yu, and K. Varadhan, "Supernetting: an Address Assignment and Aggregation Strategy," RFC 1338 (Informational), Internet Engineering Task Force, Jun. 1992.

[20] A. Liu, E. Torng, and C. Meiners, "Firewall Compressor: An Algorithm for Minimizing Firewall Policies," in *INFOCOM 2008. The 27th Conference on Computer Communications*, April 2008, pp. 176 –180.