Towards Designing Packet Filter with a Trust-Based Approach Using Bayesian Inference in Network Intrusion Detection

Yuxin Meng[1], Lam-For Kwok[1], and Wenjuan Li[2]

[1] Department of Computer Science, College of Science and Engineering, City University of Hong Kong, Hong Kong, China
ymeng8@student.cityu.edu.hk, cslfkwok@cityu.edu.hk
[2] Computer Science Division, Zhaoqing Foreign Language College, Guangdong, China
wenjuan.anastatia@gmail.com

Abstract. Network intrusion detection systems (NIDSs) have become an essential part for current network security infrastructure. However, in a large-scale network, the overhead network packets can greatly decrease the effectiveness of such detection systems by significantly increasing the processing burden of a NIDS. To mitigate this issue, we advocate that constructing a packet filter is a promising and complementary solution to reduce the workload of a NIDS, especially to reduce the burden of signature matching. We have developed a blacklist-based packet filter to help a NIDS filter out network packets and achieved positive experimental results. But the calculation of IP confidence is still a big challenge for our previous work. In this paper, we further design a packet filter with a trust-based method using Bayesian inference to calculate the IP confidence and explore its performance with a real dataset and in a network environment. We also analyze the trust-based method by comparing it with our previous weight-based method. The experimental results show that by using the trust-based calculation of IP confidence, our designed trust-based blacklist packet filter can achieve a better outcome.

Keywords: Packet Filter, IP Confidence, Trust Calculation, Network Intrusion Detection, Bayesian Inference.

1 Introduction

Over the past ten years, network intrusion detection systems (NIDSs) [1,3] have already become an important and essential component for current network security infrastructure. These detection systems are widely deployed in various network environments (e.g., a bank) to analyze network traffic and identify different kinds of network attacks (e.g., malware, spyware). Traditionally, these detection systems can be categorized into two types: signature-based NIDS and anomaly-based NIDS. The signature-based NIDS [2,4] detects an attack in terms

A.D. Keromytis and R. Di Pietro (Eds.): SecureComm 2012, LNICST 106, pp. 203–221, 2013.
© Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2013

of its signatures[1] so that this kind of detection systems can only identify well-known attacks. On the other hand, the major advantage of an anomaly-based NIDS [6,7] is the ability to detect novel attacks by means of identifying significant deviations between current network traffic and its normal profile[2]. In real deployment, a NIDS usually employs both the above two detection approaches, whereas the signature-based method is more widely used [9], compared with the anomaly-based detection, as a basis for a NIDS.

However, in a large-scale network environment, it is a big bottleneck for a NIDS, especially for a signature-based NIDS, to deal with overhead network packets. The large number of network packets can heavily consume computer resources and possibly cause a NIDS to be unable to response to current network events, which can greatly decrease the effectiveness of these detection systems [12]. Take Snort [2,8] as an example, this lightweight signature-based network intrusion detection system usually spends round about 30 percent of its total computational power in conducting signature matching between its signatures and incoming packet payloads, while its computational consumption can be significantly increased when deployed in a heavy traffic network environment. Up to 80 percent or much more of its processing burden will be put into signature matching when a massive of packets arrive [13]. Overall, its computational burden is at least linear to the size of an input packet payload [14].

In this case, these detection systems are vulnerable to *denial of service* (DoS) attacks [11,10] due to their poor performance in an intensive traffic environment. The DoS (or distributed DoS) attack is an attempt to cause a computer or network resource unavailable to its users. In the context of network intrusion detection, the DoS attack can render a detection system unusable and paralyzed, which aims to lower the level of network security by sending massive network packets to exceed the maximum processing capability of the NIDSs.

To mitigate this issue, some packet filtration mechanism has been proposed in the literature. We also advocate this approach that by appropriately filtering out a number of network packets, a network intrusion detection system can achieve more reliable and desirable performance in a large-scale network environment. But how to *appropriately* filter out network packets is still a challenge in constructing such a packet filter. In our previous work [17], we have proposed and developed an adaptive blacklist-based packet filter to filter out network packets in terms of IP confidence. Our previous approach can be treated as a reputation-based method of constructing a packet filter to address the problem of overhead network packets for a NIDS, especially for a signature-based NIDS. However, for the reputation-based method, a big suffering problem is that how to *appropriately* calculate the reputation. This issue is also a big challenge for our previous

[1] These signatures (or rules) are predefined in a NIDS and are critical to an organization to spot and remediate unwanted events in their network.

[2] Anomaly detection refers to detecting patterns in observed events that do not conform to an established normal profile. The interesting objects of this detection are often unexpected bursts in activity.

work in which we used a method of weighted ratio-based calculation to compute the IP confidence, but the calculation lacks of theoretical basis.

In this work, we aim to construct a packet filter by using a trust-based method that refers to Bayesian inference, with the purpose of enhancing the theoretical background of computing IP confidence and further improving the performance of the packet filter in a large-scale network environment. Specifically, we design a particular component called *trust calculation engine* to calculate the trust values[3] (or IP confidence) for determining the blacklisting IP addresses. The specific calculation of trust values is referred to a Bayesian inference model. We also propose that an appropriate packet filtration mechanism should have several characteristics as follows:

- The packet filter should have a minimum impact on the network performance.
- The packet filter should indeed provide a good filtration rate.
- The packet filter should not lower the whole level of network security.

The contributions of our work can be summarized in terms of the above characteristics as below:

- We further designed a trust-based blacklist packet filter by applying Bayesian inference in calculating the trustworthiness of blacklisting IP addresses. Interfering only with abnormal traffic, the impact of our packet filter on the network performance is minimum.
- In the experiment, we evaluated our approach with Snort in real settings and the experimental results showed that our packet filter could indeed help reduce the burden of a signature-based NIDS by filtering our a number of network packets (e.g., a reduction rate between 20% and 30%).
- We further analyzed the capability of our approach in defending against the DoS attack, and discussed the impact of *impersonation attacks* [15] on the packet filter. We presented that our approach would not affect and lower the whole level of network security.

In addition to the above work, we further compared our current trust-based method with our previous weight-based method in the aspect of both false rate (false positive and false negative) and traffic sensitivity by simulating network traffic in a network environment.

The rest of this paper is organized as follows. The background of our previous work is presented in Section 2; in Section 3, we show the architecture of our designed trust-based blacklist packet filter and describe the trust calculation in details; Section 4 illustrates the experimental methodology and experimental results, and we also compare the current trust-based computation with our previous weight-based calculation; Section 5 discusses the related work and we point out the future work in Section 6. Finally, we present conclusions in Section 7.

[3] The term of *trust value* is used to measure the *IP confidence*, therefore, we use these two terms interchangeably throughout this paper.

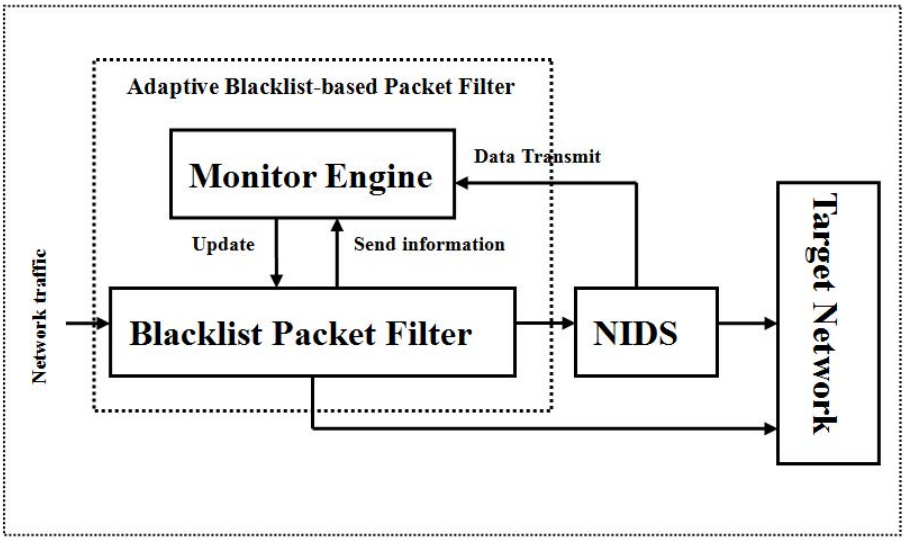

Fig. 1. The high-level architecture of the adaptive blacklist-based packet filter, which consists of a blacklist packet filter and a monitor engine

2 Background

In our previous work [17], we have proposed and developed an adaptive blacklist-based packet filter to help a NIDS filter out a number of network packets. The packet filter refines network packets depending on a blacklist, which can be generated by calculating the IP confidence. We show the high-level architecture of the adaptive blacklist-based packet filter in Fig. 1.

There are mainly two components in the *adaptive blacklist-based packet filter*: a *blacklist packet filter* and a *monitor engine*. The blacklist packet filter is responsible for filtering out network packets based on IP confidence. The monitor engine is used to collect data and update the blacklist in the *blacklist packet filter* by calculating the IP confidence.

In real deployment, this packet filter is implemented in front of the NIDS. Therefore, network traffic will firstly arrive at the *blacklist packet filter*. The filtration procedure is described as below:

- If the source IP address of a packet is not in the blacklist, then this packet will be forwarded into the NIDS for examination.
  - The NIDS examines this packet as the traditional way and decides whether to output an alarm.
  - If this packet is malicious, then the NIDS will produce an alarm and report this information to the *monitor engine*.
  - If this packet is normal, then the NIDS will send it to the target network.
- If the source IP address of a packet is in the blacklist, then this packet will be compared with the NIDS signatures stored in the *blacklist packet filter*.

- If a match is found, then the *blacklist packet filter* will generate an alarm and send a copy of this alarm to the *monitor engine*.
- If no match is identified, then the *blacklist packet filter* will directly send the packet to the target network and report the status (e.g., good or normal) of the packet (or IP address) to the *monitor engine*.

The *monitor engine* calculates the IP confidence by collecting the data from both the NIDS and the blacklist packet filter. In our previous work, we used a method of weighted ratio-based blacklist calculation. The formula of the method is shown in equation (1) (i represents the number of *good* packets, k represents the number of *bad* packets and 10 is the weighted value). In a fixed updating time, the monitor engine will update the blacklist in the *blacklist packet filter* to adapt the packet filter to the network contexts.

$$IP\ confidence = \frac{\sum_{i=1}^{n} i}{\sum_{k=1}^{m} 10 \times k} (n, m \in \mathbf{N}) \tag{1}$$

In the previous experiments, we achieved positive results that our packet filter could perform well and reduce the packets ranged from 11% to 23%. However, the IP calculation is effective and computed based on real performance. In other words, the weight-based approach of calculating the IP confidence lacks of theoretical basis. According to the work [18], the above equation is a straightforward method without the need of a distribution model, whereas it cannot accurately capture and model the uncertainty of network traffic. To improve this issue, we therefore attempt in designing our packet filter with a trust-based method of using Bayesian inference in calculating the IP confidence. Our current work aims to measure packet filtration and reduction with a theoretical model.

3 Our Proposed Method

In this section, we begin by describing the Bayesian inference and introducing its application in our designed packet filter. We then present the architecture of our further proposed trust-based blacklist packet filter and describe its components and functions in details.

3.1 Trust Value Calculation Using Bayesian Inference

In compute science, the notion of *trust* is borrowed from the social science literature aiming to evaluate and predicate the behavior of target objects. There is no clear definition for *trust* in the computer networks so that it can be interpreted as reputation, probability, trusting option, directed graphs, etc.

A lot of research work has studied and applied the notion of trust in different fields (see Section 5). In this paper, referring to some related work about IP reputation [18,19], we therefore aim to apply a trust-based method of using Bayesian inference (a theoretical model) into calculating the IP confidence for

our designed trust-based blacklist packet filter, which can greatly help the packet filter deal with variants in network traffic.

As discussed in Section 2, the adaptive blacklist-based packet filter can be regarded as a reputation-based method. Thus, we can calculate the trust values (or IP confidence) by applying other trust-based approaches. In statistics, *Bayesian inference* is a method of inference in which Bayes' rule is used to update the probability estimate for a hypothesis as additional evidence. The objective of using the *Bayesian inference* in our work is to determine whether an IP address should be blacklisted. We give a major assumption as follows:

– *Assumption.* We assume that all packets are independent from each other. That is, if one packet is found to be a malicious packet, the possibility of the following packet being a malicious packet is still 1/2.

This possibility assumption indicates that the attacks may come in various forms, either in one packet or in a number of packets. To derive the calculation of trust values. We assume that N packets are sent from an IP address to the trust-based blacklist packet filter, of which k packets are proven to be *normal*. we further provide some terms as below:

$$V_i \ \ (means\ that\ the\ i^{th}\ packet\ is\ normal.)$$

$$n(N) \ \ (means\ the\ number\ of\ normal\ packets.)$$

$$P(n_i : normal) = p\ (means\ the\ possibility\ of\ the\ i^{th}\ incoming\ packet\ is\ normal.)$$

In terms of the work [18,19] and the above assumption, the distribution of observing $n(N) = k$ is governed by Binomial distribution[4] as below.

$$P(n(N) = k|p) = \binom{N}{k} p^k (1-p)^{N-k} \tag{2}$$

Then, our objective is to estimate the possibility $P(V_{N+1} = 1|n(N) = k)$. We can use the *Bayesian Inference* approach to calculate this possibility. From Bayesian equation, we can have the following probability distribution.

$$P(V_{N+1} = 1|n(N) = k) = \frac{P(V_{N+1} = 1, n(N) = k)}{P(n(N) = k)} \tag{3}$$

For the above equation, we use marginal probability distribution[5] and have:

$$P(n(N) = k) = \int_0^1 P(n(N) = k|p) f(p) \cdot dp \tag{4}$$

$$P(V_{N+1} = 1, n(N) = k) = \int_0^1 P(n(N) = k|p) f(p) p \cdot dp \tag{5}$$

[4] Binomial distribution is the discrete probability distribution that represents the number of successes in a sequence of n independent, which the possibility of each n is the same p.

[5] Marginal distribution of a subset of random variables is the probability distribution of the variables contained in the subset.

There is no prior information about p, so that we assume that p is determined by a uniform prior distribution $f(p) = 1$ where $p \in [0, 1]$. Therefore, using equation (2), (3), (4) and (5), we can have the following equation:

$$P(V_{N+1} = 1|n(N) = k) = \frac{\int_0^1 P(n(N) = k|p)f(p)p \cdot dp}{\int_0^1 P(n(N) = k|p)f(p) \cdot dp} = \frac{k+1}{N+2} \quad (6)$$

Based on the equation (6), we can calculate the trust values (denoted t_{value}) for relevant IP addresses. If we set a threshold to $T \in [a, b]$ (the selection of the *threshold* will be discussed later), then we can judge a blacklisting IP address[6] to be maintained or deleted as follows:

- If $t_{value} \in T$, then the blacklisting IP address will be deleted from the blacklist.
- If t_{value} is not in T, then the blacklisting IP address will be still in the blacklist.

3.2 Architecture of Trust-Based Blacklist Packet Filter

As shown in Fig. 2, we describe the architecture of our further designed *trust-based blacklist packet filter*. There are totally two major components: a *blacklist*

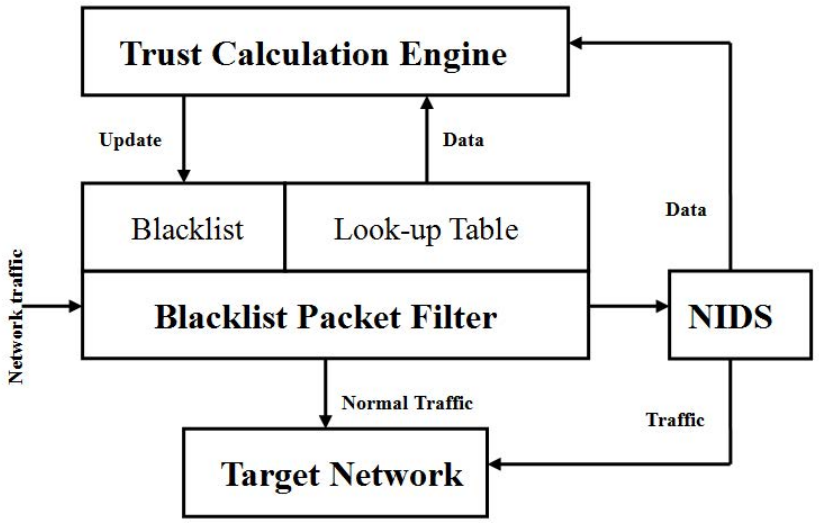

Fig. 2. The architecture of the trust-based blacklist packet filter, which consists of a blacklist packet filter and a trust calculation engine

[6] In the packet filter, a new IP address will be blacklisted as long as a NIDS alarm for this IP address is produced.

packet filter and a *trust calculation engine*. The blacklist packet filter is similar in our previous work and is mainly responsible for filtering out network packets based on trust values. It consists of two components: a *blacklist* and a *look-up table*. The blacklist contains all blacklisting IP addresses while the look-up table contains NIDS signatures indexed by the blacklisting IP addresses. The *trust calculation engine* is used to collect data from both the blacklist packet filter and the NIDS, and is responsible for computing trust values and updating the blacklist accordingly. When network packets arrive, the filtration procedure of the *trust-based blacklist packet filter* is described as below:

- If the IP address of a packet is in the *blacklist*, then the packet payload will be compared with the signatures stored in the *look-up table*.
  - If a match is found, then the *blacklist packet filter* will produce an alarm and send a copy of this alarm to the *trust calculation engine*.
  - If no match is found, then the packet will be sent to the target network.
- If the IP address of a packet is not in the *blacklist*, then the packet will be forwarded into the NIDS for examination

In Fig. 3, we give the construction of the *look-up table* in the blacklist packet filter. The *look-up table* contains two sub-tables: *table of Matched NIDS Signatures* and *table of All NIDS Signatures*. The *table of All NIDS Signatures* contains all NIDS signatures that are active in the NIDS signature database. The *table of Matched NIDS Signatures* contains the NIDS signatures that have been matched in the detection procedure and the matched NIDS signatures are indexed by blacklisting IP addresses. The comparison procedure in the *look-up table* is described as below:

For a payload from an IP address, the look-up table will firstly search in the *table of Matched NIDS Signatures* based on its IP address.

- *Situation1.* For this IP address, if there are no any signatures in the *table of Matched NIDS Signatures*, then the *look-up table* will compare the payload with the signatures in the *table of All NIDS Signatures*.
  - If a match is found, then an alarm will be produced.
  - If no match is found, then the packet will be sent to the target network.
- *Situation2.* For this IP address, if there are signatures existing in the *table of Matched NIDS Signatures*, then the *look-up table* will compare the payload with the matched signatures.
  - If a match is found, then an alarm will be produced.
  - If no match is found, then the *look-up table* will compare the payload with all signatures in the *table of All NIDS Signatures*. The comparison process is the same as *Situation1*.

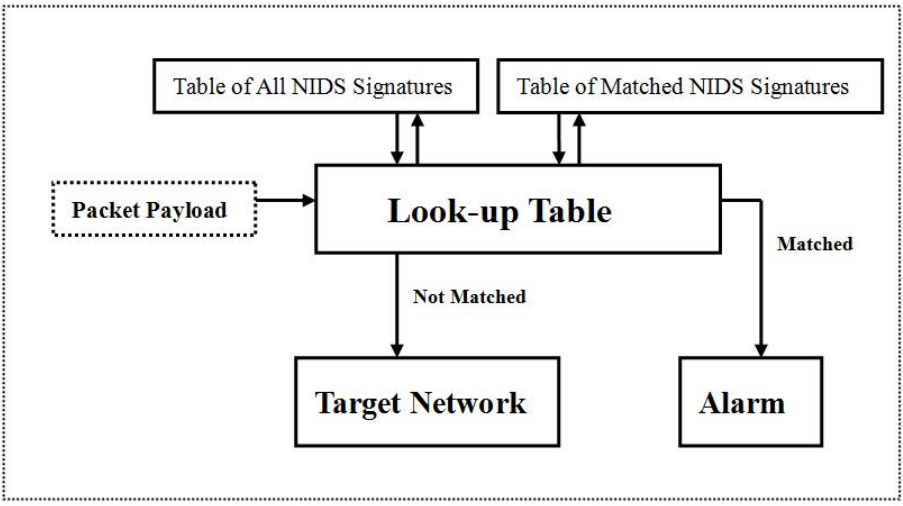

Fig. 3. The construction of the look-up table: table of Matched NIDS Signatures and table of All NIDS Signatues

4 Evaluation

To evaluate the trust-based approach, in this section, we describe the experimental methodology, illustrate how to determine the threshold in the designed packet filter, present experimental results and discuss the current approach with our previously used weight-based method.

4.1 Experimental Methodology

The first question is that how to set an appropriate threshold for distinguishing normal and abnormal IP addresses. According to the equation (6), we can calculate the trust values (t_{value}) as follows:

$$t_{value} = \frac{k+1}{N+2}$$

Therefore, if k is big enough which means that normal packets dominate the network traffic, then the t_{value} will become larger. Since k is smaller than N (the total number of packets), the value range of t_{value} is belonging to [0,1]. In this case, the best scenario for t_{value} is that its value infinitely close to 1, which means that the vast majority of current network packets are normal. On the other hand, when the t_{value} declines, it means that malicious packets are detected in the network environment. Therefore, the threshold can be initially presented as [a,1]. To determine the lower limit a of the threshold, we simulate some normal traffic to the trust-based blacklist packet filter and identify the threshold by analyzing the simulation results.

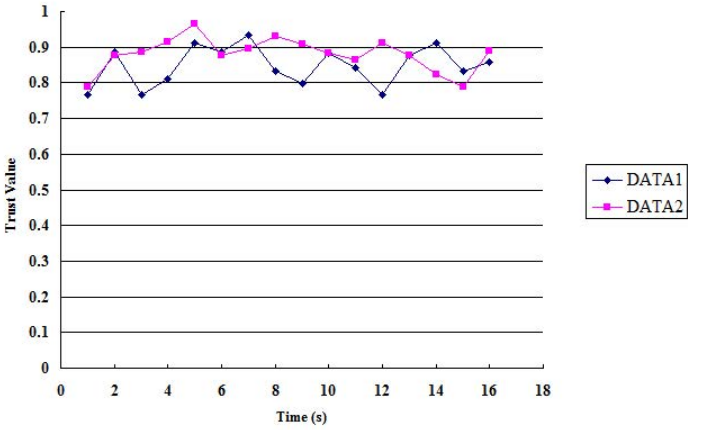

Fig. 4. The average trust values for the two parts of the real dataset: *DATA1* and *DATA2*

After obtaining the threshold, we then investigate the performance of our trust-based blacklist packet filter with a real dataset and in a network environment, comparing with the performance of Snort. At last, in the network environment, we compare the calculation of trust values with the weighted ratio-based calculation by simulating some normal and malicious packets.

4.2 Threshold Selection

In order to select an appropriate lower limit for the threshold, we conducted an experiment for the designed packet filter by using a real dataset. The real dataset was captured by a Honeypot[7] which was deployed in our CSLab. The Honeypot provided several services (e.g., FTP, HTTP) for users from outside network and recorded all incoming traffic. The incoming traffic can contain both normal and abnormal traffic.

By analyzing the captured traffic, we constructed a real dataset and divided it into two parts (called *DATA1* and *DATA2*), with about 4 to 6 million packets and the base rates are nearly B=0.003325 and B=0.001723 respectively which are regarded to be reasonable and normal in real settings. We simulated the traffic to our packet filter and the results are shown in Fig. 4.

In the experiment, the trust values will be updated in every 1 second. The average trust values are simply average values of all IP addresses in the dataset. For the *DATA1*, its average trust values are from 0.765 to 0.934, while for the *DATA2*, the average trust values are from 0.788 to 0.965. On the whole, the range of trust values is between 0.75 and 1.0. Therefore, based on the simulation results, we select the threshold to [0.75,1].

[7] This project is managed by HoneybirdHK (`http://www.honeybird.hk/`)

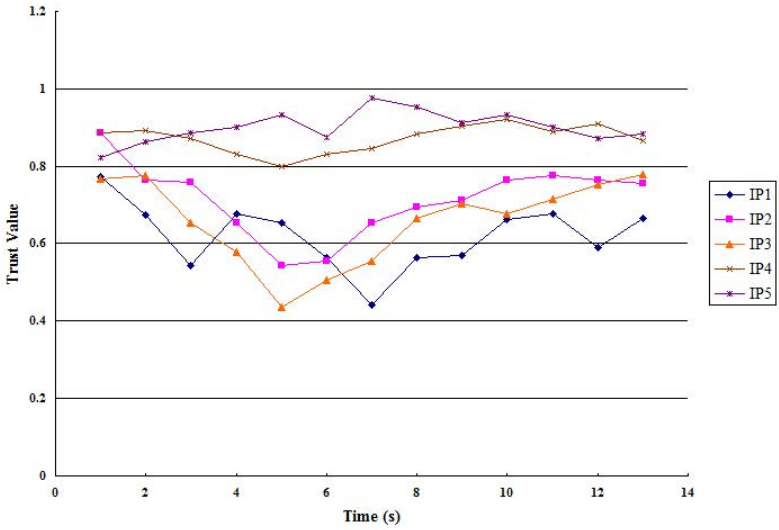

Fig. 5. The trust values for *IP1*, *IP2*, *IP3*, *IP4*, *IP5* in *DATA3*

4.3 Experiment with Real Dataset

Based on the Honeypot, we additionally constructed another dataset called *DATA3* to explore the initial performance of the trust-based blacklist packet filter. By analyzing the *DATA3* in advance, we have found some malicious packets are from the IP addresses: denoted *IP1*, *IP2* and *IP3*. We present the trust values about these possible blacklisting IP addresses in Fig. 5.

The trust values will be updated in every 1 second. It is visible that the trust values for *IP1*, *IP2* and *IP3* gradually decline below the threshold [0.75,1] when these IP addresses send some malicious packets. In comparison, we give the trust values of two normal IP addresses: *IP4*, *IP5*. As shown in Fig. 5, the trust values of these two normal IP addresses steadily fall within the threshold [0.75,1]. The results of this experiment show that the trust-based blacklist packet filter is capable of detecting the malicious IP addresses that are sending malicious packets mixing with normal packets.

In the experiment, it is hard for the trust values to reach the perfect value 1 since packet record may arrive late and the *trust calculation engine* will not count these packets in the calculation of trust values. That is, the *trust calculation engine* may not consider the late packets to be normal packets in nature. This mechanism ensures that only confirmed normal packets can be used in calculating the trust values, which can secure the trust calculation.

4.4 Experiment in a Network Environment

To further investigate the performance of the packet filter in the aspect of packet filtration, we constructed a network environment by using existing tools and

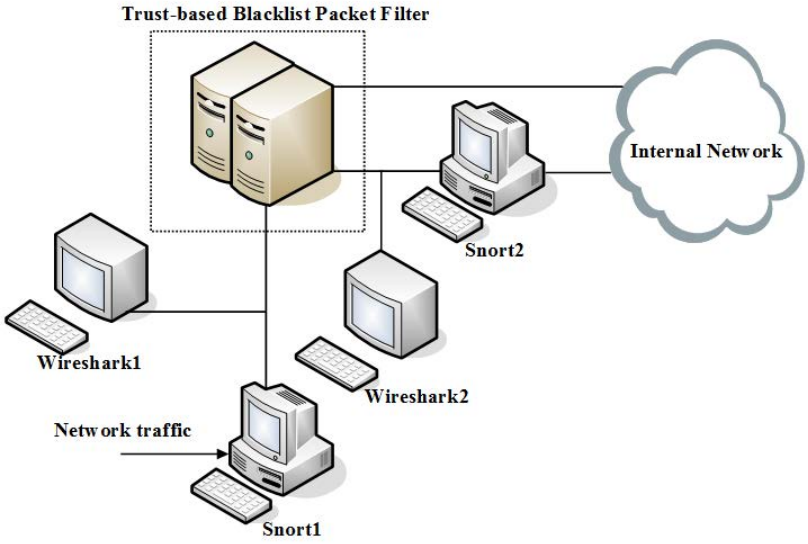

Fig. 6. The experimental deployment consists of Snort1, Wireshark1, Wireshark2, trust-based blacklist packet filter, Snort2 and Internal Network

deployed the trust-based blacklist packet filter in this network environment. The experimental deployment is shown in Fig. 6.

The network environment mainly consists of Snort, Wireshark [16] and the trust-based blacklist packet filter. In particular, we implemented two Snort in the network environment, one (named *Snort1*) is deployed in front of the *trust-based blacklist packet filter* whereas the other (named *Snort2*) is deployed behind the packet filter. Due to this deployment, we can evaluate the capability of the packet filter in reducing the burden of a NIDS by comparing the performance between *Snort1* and *Snort2*. The Wireshark is responsible for monitoring network packets and verifying the performance of our packet filter in the aspect of packet reduction by analyzing recorded packet information.

We conducted the experiment for a week and the first-day results of CPU usage between *Snort1* and *Snort2* are presented in Fig. 7. The results show that the CPU usage of *Snort1* generally larger than that of *Snort2* by implementing in the same network environment. The CPU-usage performance of other 6 days is similar to the first day, which means that our packet filter can indeed reduce the burden of a NIDS by filtering out a number of network packets. In Table 1, we show the packet reduction rate for 7 days. The information is calculated based on the recorded data from the two Wireshark tools. It is easily visible that our packet filter can achieve a packet reduction rate in the range from 21.54% to 33.87% in the experimental network environment. The results verify that our packet filter is able to filter out network packets by using the trust-based approach to calculate the IP reputation.

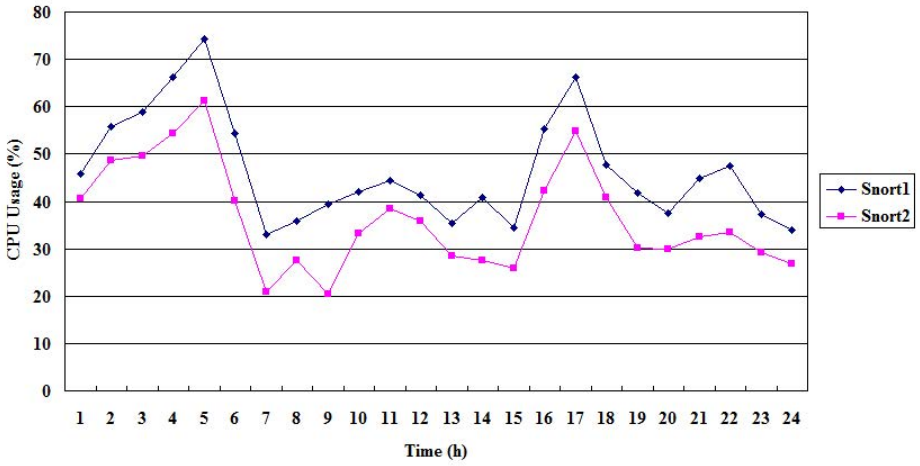

Fig. 7. The CPU usage of *Snort1* and *Snort2* for the first day

Table 1. Results of Packet Reduction Rate

Week Day	*Packet Reduction Rate (%)*	**Week Day**	*Packet Reduction Rate (%)*
Monday	21.54	Saturday	24.33
Tuesday	22.56	Sunday	25.80
Wednesday	31,67		
Thursday	27.84		
Friday	33.87		

The specific packet reduction rate is depending on the number of blacklisting IP addresses in the *blacklist packet filter*. In general, more IP addresses are blacklisted, bigger reduction rate can be achieved. In this case, the packet reduction rate in a real network environment may be fluctuant in terms of network contexts (i.e., when the network traffic is becoming normal, the reduction rate will be decreased, but if the network traffic contains a lot of malicious packets, then the reduction rate will be possibly increased). More future experiments can be conducted to explore this relationship.

4.5 Outcome Comparison

The above experiments show positive results of our designed trust-based blacklist packet filter in reducing the burden of a NIDS by filtering out network packets. In this section, we compare the trust-based approach with our previous weight-based method in the aspect of blacklist generation.

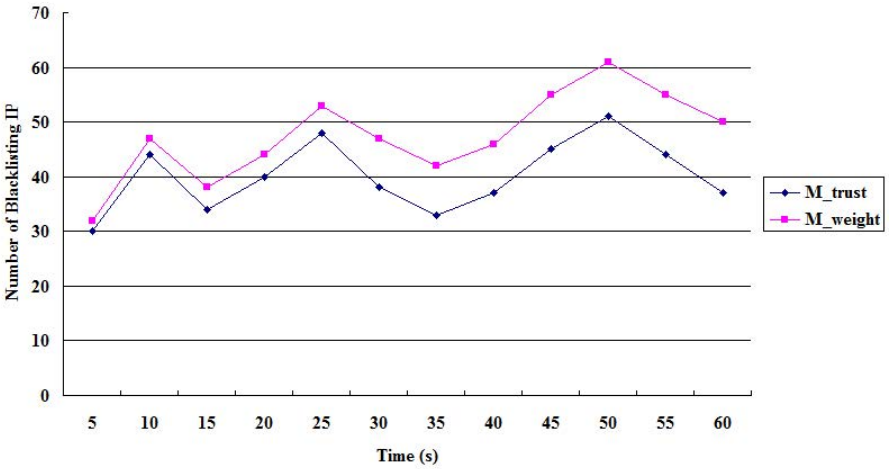

Fig. 8. The number of blacklisting IP addresses for M_{trust} and M_{weight}

Our packet filter is based on a blacklist, thus, it is very important to appropriately generate a *good* blacklist. The meaning of a *good* blacklist can be represented as follows:

- The blacklist should accurately reflect the current traffic. In other words, the false positive and the false negative of the blacklist should be low.
- The blacklist should be sensitive to the traffic change. That is, when a potential malicious IP address is detected or deleted, the blacklist should contain or remove this IP address in terms of calculated trust values or IP confidence.

To compare the two approaches (we denote our current approach as M_{trust} while our previous method as M_{weight}), we deployed these approaches in a network environment like Fig. 6 and simulated some traffic to both mechanism. During the experiment, we utilized a packet generator [5] to simulate some malicious IP addresses by sending out some malicious packets. The number of blacklisting IP addresses for both methods is shown in Fig. 8.

At the beginning, we simulated 33 malicious IP addresses. The approach of M_{trust} blacklisted 30 of them whereas the approach of M_{weight} blacklisted 32 of them. The detection rate of M_{weight} is a bit higher since we use a 10-weighted ratio based method to emphasize the impact of every malicious packet.

Then in the time interval of 5s to 10s, we additionally simulated 15 new malicious IP addresses. For the M_{trust}, it blacklisted all these new IP addresses while the number of blacklisting IP is 44 rather than 45, the reason is that 1 blacklisting IP address has become normal in terms of its trust value. For the M_{weight}, it detects all these new malicious IP with no blacklisting IP becoming normal. Subsequently, we only maintained 32 malicious IP addresses to send malicious packets between 10s and 15s. It is easily visible that M_{trust} can quickly adaptive to this change and its number of blacklisting IP addresses decreases to

34. But for M_{weight}, its number of blacklisting IP addresses only decreases from 47 to 38. During [25s,30s], [30s,35s] and [50s, 60s], we maintained the number of malicious IP addresses to 36, 30 and 34 respectively. Similarly, we find that M_{trust} is more sensitive to the traffic changes than M_{weight}.

Overall, based on the simulation results, both of the two approaches have an acceptable false positive and false negative (i.e., M_{trust} with FN 6.8% and FP 8.32%, M_{weight} with FN 2.2% and FP 15.4%). The false positive of M_{weight} is higher than M_{trust} in that we use 10 as the weighted value in calculating IP confidence which means that an IP address may be blacklisted by sending only several malicious packets. On the other hand, due to the weighted blacklist generation, M_{weight} is more powerful in detecting a malicious IP address. On the whole, the false positive and the false negative of both approaches are acceptable. Regarding to the sensitivity, the approach of M_{trust} is greatly more sensitive to the traffic changes in a network than M_{weight}. Based on the definition of a *good blacklist*, we consider M_{trust} is generally better than M_{weight} by considering both *false rate* (false positive and false negative) and *traffic sensitivity*.

4.6 Security Discussion and Potential Countermeasures

DoS Attack. As discussed before, DoS attack is a big problem for a NIDS. By implementing the packet filter, a lot of network packets can be filtered out so that the possibility of a NIDS surviving in a large-scale network environment will be increased.

For the packet filter, DoS attack is also a big challenge as for other packet filters that some countermeasures should be considered. However, the countermeasures should not affect the network security too much. We therefore consider employing a *d-threshold* into our packet filter that all packets from an IP address will be discarded if the trust value of this IP address is below the *d-threshold*. In this case, the possibility range [0,1] can be further divided into three intervals:

- *[0,d-threshold]*. When the trust values belong to this interval, all packets from these IP addresses will be discarded.
- *[d-threshold,0.75]*. When the trust values fall in this interval, all packets from these IP addresses will still be compared with NIDS signatures by the trust-based blacklist packet filter in order to keep the level of network security.
- *[0.75,1]*. When the trust values are classified into this interval, all packets will be processed into a NIDS for examination.

The DoS attack can be partly mitigated by employing a *d-threshold*. If the trust value of an IP address is smaller than this *d-threshold*, it means that this IP address is harmful to the network. Therefore, it is crucial to appropriately select this *d-threshold*. Further experiments should be conducted to collect more data to investigate this issue.

IP Spoofing. This IP spoofing attack is a kind of impersonation attacks, which refers to sending network packets by concealing the identity of the sender or

impersonating another computer users. The final goal of this attack is possibly to launch a DoS attack, which affects the availability of network resources.

For our packet filter, the IP spoofing attack may succeed in bypassing the filtration of the packet filter. However, as discussed in our previous work [17], this attack will not affect the whole level of network security since the packets still need to be examined by a NIDS even if these packets bypass our packet filter. Moreover, our packet filter and the NIDS use the same NIDS signature database so that the detection capabilities of the packet filter and the NIDS are the same. To further mitigate this attack, we can develop an IP verification mechanism to verify the IP source and filter out spoofed packets. More experiments and data should be collected to evaluate this approach.

5 Related Work

Trust-based methods have been applied in many fields. Gonzalez *et al.* [19] presented a work by using Bayesian inference in defending against IP spoofing attacks at the router level. Their results showed that their application could effectively detect malicious access routers and has a low impact on the network performance. Our work is different from their work in that we apply the Bayesian inference and Bayesian model into network packet filtration to help compute IP confidence (determine blacklist) and construct a trust-based blacklist packet filter. It is visible, from our work, that the trust-based method is a promising method that can be applied into the evaluation of packet filtration. To the best of our knowledge, our work is an early work that attempts in designing a packet filter with a Bayesian model and applying this probability model into producing a blacklist. We expect to see more work to be done in this research area.

For the application of trust-based approaches, Yao *et al.* [20] proposed a Bayesian network-based trust model for a peer-to-peer file sharing application, which could present differentiated trust and combine different aspects of trust. Sun *et al.* [18] presented an information theoretical framework to quantitatively measure trust and to build a model for trust propagation in ad hoc networks. The framework was developed to secure ad hoc routing and malicious node detection. Then, Zhu *et al.* [24] extended the above idea to formalize the trusted actions by using mutual information to quantify trust and to use MaxMin mechanism to calculate trust which could be established through multiple recommendation paths in ad-hoc networks. Later, Chung *et al.* [21] presented a trust model, based on Bayesian networks, which could adapt to ad hoc networks and distributed systems. Their model evaluated the trust in a server based on two points: direct experiences with the server and recommendations concerning its service.

For filtering out packets in intrusion detection, Ioannis *et al.* [22] introduced a packet pre-filtering approach, which was a powerful hardware-based technique, as a means to resolve the burden of an intrusion detection system. They implemented the header matching portion of a NIDS system together with a small prefix match that the rules could be checked more efficiently by a full-match

module. Later, Ning *et al.* [23] proposed a high-performance memory-based IDS that could be easily reconfigured for new rules by utilizing deep packet prefiltering and novel finite state encoding.

6 Future Work

A lot of studies have been conducted on constructing packet filters. But it is still a hot topic for efficiently designing such kind of filters and *appropriately* evaluating the packet filtration and reduction. Our current work aims to design a packet filter to adaptively filter out network packets by calculating IP confidence and generating a blacklist with a theoretical model. There are many possible work in future experiments. The future work could include exploring the performance of the trust-based blacklist packet filter in a distributed network environment (i.e., exploring whether the threshold is the same when deployed in a distributed network environment). Future work could also include employing more information theory (e.g., entropy theory) in calculating the IP confidence and evaluating the performance of packet filtration and reduction.

7 Conclusion

The performance of a network intrusion detection system is greatly restricted in a large-scale network environment. That is, overhead network packet can significantly reduce the effectiveness of a NIDS and heavily consume computer and network resources. To mitigate this issue, we advocate that constructing a packet filter is a promising solution.

In this work, we further design a *trust-based blacklist packet filter* to reduce the burden of a NIDS by filtering out a number of network packets. Specifically, the trust-based blacklist packet filter consists of two major components: a *blacklist packet filter* and a *trust calculation engine*. The *blacklist packet filter* is responsible for filtering out network packets in terms of IP confidence while the *trust calculation engine* is responsible for collecting data and updating the blacklist. The blacklist is generated by computing the trust values (or IP confidence) by using a trust-based approach of Bayesian inference.

In the experiment, we showed how to select an appropriate threshold for our packet filter. We then evaluated the performance of the packet filter with a real dataset and in a network environment. The experimental results show that the packet filter is effective at filtering out network packets without lowering the network security and has a minimum impact on the network performance. We further compared our current trust-based method with our previous weight-based method and the simulation results describe that the trust-based method is generally better by considering both false rate and traffic sensitivity.

Acknowledgments. We would like to thank HoneybirdHK for supporting and providing the real dataset and all anonymous reviewers for their valuable comments.

References

1. P.V.: Bro: A System for Detecting Network Intruders in Real-Time. Computer Networks 31(23-24), 2435–2463 (1999)
2. Roesch, M.: Snort: Lightweight Intrusion Detection for Networks. In: 13th Large Installation System Administration Conference (LISA), pp. 229–238. USENIX Association Berkeley, CA (1999)
3. Scarfone, K., Mell, P.: Guide to Intrusion Detection and Prevention Systems (IDPS). NIST Special Publication 800-94 (February 2007)
4. Vigna, G., Kemmerer, R.A.: NetSTAT: A Network-based Intrusion Detection Approach. In: Annual Computer Security Applications Conference (ACSAC), pp. 25–34. IEEE Press, New York (1998)
5. Colasoft Packet Builder, `http://www.colasoft.com`
6. Valdes, A., Anderson, D.: Statistical Methods for Computer Usage Anomaly Detection Using NIDES. Technical Report, SRI International (January 1995)
7. Ghosh, A.K., Wanken, J., Charron, F.: Detecting Anomalous and Unknown Intrusions Against Programs. In: Annual Computer Security Applications Conference (ACSAC), pp. 259–267 (1998)
8. Snort, The Open Source Network Intrusion Detection System, `http://www.snort.org/`
9. Sommer, R., Paxson, V.: Outside the closed world: On using Machine Learning for Network Intrusion Detection. In: IEEE Symposium on Security and Privacy, pp. 305–316. IEEE, New York (2010)
10. Carl, G., Kesidis, G., Brooks, R.R., Suresh, R.: Denial-of-Service Attack-Detection Techniques. IEEE Internet Computing 10(1), 82–89 (2006)
11. Paxson, V.: An Analysis of using Reflectors for Distributed Denial-of-Service Attacks. ACM Computer Communication Review 31(3) (July 2001)
12. Dreger, H., Feldmann, A., Paxson, V., Sommer, R.: Operational Experiences with High-volume Network Intrusion Detection. In: ACM Conference on Computer and Communications Security (CCS), pp. 2–11. ACM, USA (2004)
13. Fisk, M., Varghese, G.: An Analysis of Fast String Matching Applied to Content-based Forwarding and Intrusion Detection. Technical Report CS2001-0670, University of California, San Diego (2002)
14. Rivest, R.L.: On the Worst-case Behavior of String-Searching Algorithms. SIAM Journal on Computing 6, 669–674 (1977)
15. Michel, B., Jyanthi, H., Evangelos, K.: Detecting Impersonation Attacks in Future Wireless and Mobile Networks. In: Workshop on Secure Mobile Ad-hoc Networks and Sensors, pp. 1–16 (2005)
16. Wireshark, `http://www.wireshark.org/`
17. Meng, Y., Kwok, L.F.: Adaptive Context-aware Packet Filter Scheme using Statistic-based Blacklist Generation in Network Intrusion Detection. In: 7th International Conference on Information Assurance and Security (IAS 2011), pp. 74–79. IEEE Press, New York (2011)
18. Sun, Y., Yu, W., Han, Z., Liu, K.: Information Theoretic Framework of Trust Modeling and Evaluation for ad hoc Networks. IEEE Journal on Selected Areas in Communications 24(2), 305–317 (2006)
19. Gonzalez, J.M., Anwar, M., Joshi, J.B.D.: A Trust-based Approach against IP-Spoofing Attacks. In: 9th International Conference on Privacy, Security and Trust (PST 2011), pp. 63–70 (2011)

20. Yao, W., Julita, V.: Bayesian Network-Based Trust Model. In: IEEE/WIC International Conference on Web Intelligence, pp. 372–378. IEEE, New York (2003)
21. Chung, T.N., Camp, O., Loiseau, S.: A Bayesian Network based Trust Model for Improving Collaboration in Mobile ad hoc Networks. In: IEEE International Conference on Research, Innovation and Vision for the Future, pp. 144–151 (2007)
22. Ioannis, S., Vasilis, D., Dionisios, P., Stamatis, V.: Packet Pre-filtering for Network Intrusion Detection. In: ACM/IEEE Symposium on Architecture for Networking and Communications Systems (ANCS), pp. 183–192. ACM, New York (2006)
23. Ning, W., Luke, V., Benfano, S.: Deep Packet Pre-filtering and Finite State Encoding for Adaptive Intrusion Detection System. Computer Networks 55(8), 1648–1661 (2011)
24. Zhu, H., Bao, F.: Quantifying Trust Metrics of Recommendation Systems in Ad-Hoc Networks. In: 2007 IEEE Wireless Communications and Networking Conference (WCNC), pp. 2904–2908. IEEE, New York (2007)