جامعة نايف العربية للعلوم الأمنية
Naif Arab University for Security Sciences

Naif Arab University for Security Sciences
Journal of Information Security & Cybercrimes Research
مجلة بحوث أمن المعلومات والجرائم السيبرانية
https://journals.nauss.edu.sa/index.php/JISCR

JISCR

Survey on Updating IDSs' Signatures Databases

Muteb Y. Al-Yosef [1*], Nabih T. Arar[2]
[1] *T Department, Saudi Civil Defense, Riyadh, Saudi Arabia.*
[2] *Department of Network Security, College of Computer and Information Security, Naif Arab University for Security Sciences, Riyadh, Saudi Arabia.*

Received 29 Mar. 2018; Accepted 10 May 2018; Available Online 30 May 2018.

Abstract

With the rapid development and growth of the internet and networking, greater numbers of attacks are arising that threaten networks and information security alike. Thus, different types of intrusion detection systems (IDSs) have been introduced: either signature-based IDSs, anomaly-based IDSs, or a hybrid of both. Many IDSs that have adopted the signature-based method suffer from many challenges, one of these challenges is how to detect a new attack in the incoming traffic that its signature doesn't stored in the known signatures database, while at the same time keeping the rate of false-positive alarms low. Many IDSs update their signatures databases from time to time through the internet or by relying on the network administrator to manually update the database with new attack signatures. Manual updating is a labour-intensive process, can be prone to errors, and is not always practical. This is a survey paper on the various studies regarding the updating process for known IDSs' signatures databases over time.

I. Introduction

The acceleration of intrusion throughput with intelligent codes and changeable signature patterns resulting from the extensive use of high-speed networks in all transactions may lead to violating the confidentiality, integrity, and availability of security policies of computers and networks [1]. Thus, IDSs are needed to detect and protect networks as well as their hosts from these possible attacks, as it is considered the second defender after the firewall. The use of a most reliable IDS is important because building a fully secure system is almost impossible [2]. The detection process will be a difficult or impossible task if the new attack signature cannot be added to the IDS's signatures database as quickly as it is created. Otherwise, the IDS will be useless. As an example, in 2009, Rao et al. [3] proposed a new architecture that included an updater engine to retrieve the new attack signatures from the master database, connected to the cloud, in order to update the local database, connected to the IDS.

In general, the IDS is categorised into two types: signature-based and behaviour-based detection [4]. In signature-based detection, there are a number of predefined signatures of known attacks. The incoming packet is matched among these predefined stored signatures. Once it is matched with one of the stored signatures, the alarm is generated. This method is effective in detecting known attacks, but it cannot detect

Keywords: *intrusion detection system, signature-based, anomaly-based, traffic, false positive, update*

Production and hosting by NAUSS

* Corresponding Author: Muteb Y. Al-Yosef
Email: 4360864@nauss.edu.sa
doi: 10.26735/16587790.2018.005

new attacks whose signatures are not stored in the known signatures database [4]. However, behaviour-based (or anomaly-based) detection identifies the profile of normal behaviour. Any deviation from these predefined normalities will result in generating an alarm. This method is effective in detecting unknown attacks, but it suffers from a large number of false alarms [5].

The rest of this survey is organised as follows. The next section presents the survey's objectives. Then, related work on previous studies about signatures database updating were shown. The final section presents the conclusion.

II. Objectives

This survey mainly focuses on previous studies about the working mechanisms, techniques, and methods used to update the signatures database of any IDS tool that adopts the signature-based approach to detecting attacks and how these techniques work, as well as the defects and gaps in these techniques. The survey then analyses the final results of the previous studies to determine how to strengthen future work points regarding the updating process.

III. Related Work

In their paper, Salour and Su [6] were motivated by the fact that a small signature database can improve the performance of a signature-based IDS since a packet needs to match fewer signatures. This is because, when an IDS faces a huge amount of network traffic flood that exceeds its potentials, all it can do is drop packets and, therefore, may miss dangerous attacks.

Based on that fact, Salour and Su [6] proposed a dynamic model for a two-layer signature-based IDS with unequal databases that can detect imminent threats with a very high success rate by automatically

creating and using a small efficient database in the primary IDS and, at the same time, detect all other known threats by using a less efficient complementary database in the secondary IDS through automating the signatures update between the two IDSs. The model consists of two IDSs deployed in two layers: a primary IDS containing a database of the most frequent attack signatures and a complementary IDS with a large database containing the remaining signatures recognisable by the system during the training process.

The model concentrates on the packets' payload and identifies whether it carries an attack or not. Salour and Su [6] presented an automatic way to decide the set of rules and signatures to be deployed in the two IDSs and continuously updates the rules based on the usage pattern. Next, the most frequent attacks in the training process are identified based on some factors: the minimum frequency rate at which a signature occurs, the age of the alert in the database, and the maximum number of signatures that the user would like to keep in the primary IDS. Then, a signature database is created for the primary IDS that contains signatures for the most frequent attacks, and a complementary signature database is created for the secondary IDS containing the remaining signatures. After the initial training period, the secondary IDS, during a certain time, checks to see if any signature occurs more than the determined frequency threshold; if so, that signature will be removed from the secondary database and added to the primary IDS signature database. This is called the automatic signature update process, as illustrated in Fig. 1.

At the same time, the primary IDS, also on certain intervals, checks to see if any signature in its own database is no longer being seen frequently so it can update its own signature database and the secondary IDS's signatures database simultaneously, as illustrated in Fig. 2.

Salour and Su [6] developed a C program as an engine to create the most frequent signatures database for the primary IDS and a complementary signatures database for the secondary IDS on certain intervals to keep the

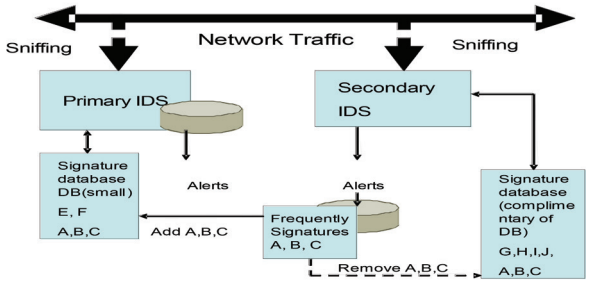

Fig. 1. Automatic signature update on secondary IDS.

frequent signatures database and the complementary signatures database constantly updated, as illustrated in Algorithm 1.

The experiment shows the difference in the performance of the IDS using a small signatures database, compared to just enabling all signatures, as shown in Table I.

Salour and Su [6] proved in their experiment that there is a significant decrease in the packet drop rate, even with high traffic flood, and as a result, there is a significant improvement in detecting threats to the network. Furthermore, they advised extending their proposed model into a multilayer IDS with a dynamic

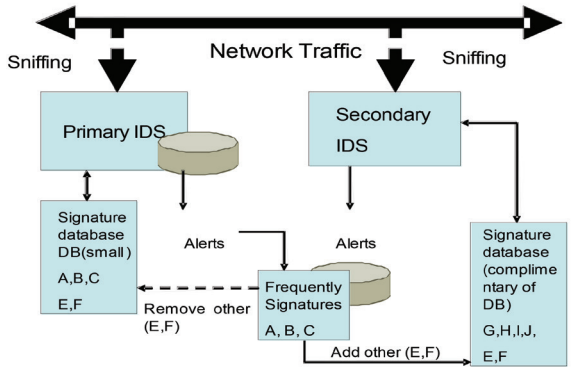

Fig. 2. Automatic signature update on primary IDS.

Algorithm 1: Generate and Update Signatures Databases.

1. N=0 # number of current signatures
2. Query the MySQL database to retrieve the set of signatures detected, S.
3. **for** every signature f in S **do**
4. Freq = number of occurrences of f
5. LTime = last detection time of f
6. **if** N <= MaxNum and Freq >= MinFreq and Ltime >= ValidTime **then**
7. remove the signature from the secondary database.
8. Add the signature in the primary IDS
9. N = N + 1
10. **End if**
11. **End for**
12. restart primary and secondary IDS

TABLE I
TESTING THE PERFORMANCE OF IDS USING BOTH SMALL AND BIG SIGNATURES DATABASES.

	Scenario 1 (All rules enabled)	**Scenario 2 (Most frequent signatures enabled)**
Test 1		
Number of packets received	187436	237503
Number of packets analyzed	173039	236700
Number of packets dropped by Snort	14397	803
Percentage of packets dropped	**7.618%**	**0.338%**
Test 2		
Number of packets received	182688	193890
Number of packets analyzed	167533	190581
Number of packets dropped by Snort	15155	3309
Percentage of packets dropped	**8.296%**	**1.707%**
Test 3		
Number of packets received	211043	241856
Number of packets analyzed	188997	237369
Number of packets dropped by Snort	22046	4487
Number of packets dropped by Snort	**10.446%**	**1.855%**

update engine to add and remove the signatures in databases of multiple IDSs based on many criteria, such as the frequency of signatures' appearances and the age of the alert in the database.

Salour and Su [6] performed their work by distributing the signatures into two levels with unequal databases. The primary level contains a small database with the most frequent signatures based on predefined factors, and the secondary level contains a large database

with all the remaining signatures. The signatures update process occurs automatically between the two databases at certain intervals defined by the user. Their model lacks the ability to update the large database dynamically without the administrator's interference to detect a new attack that does not appear during the training process and whose signatures are not stored in both databases. Therefore, they proved that working on two unequal updated databases is better than working on one large database in order to avoid the packets dropping and Distributed Denial of services (DDOS) attacks.

Rao et al. [6] proposed an architecture that is suitable for any organisation with multiple sites with IDS deployed for each, where each site can have a predefined set of rules customised based on the site's needs and policies. The proposed architecture can manage the distributed system components efficiently, as shown in the Fig. 3.

The proposed solution contains a cache component that collects network packets. The sampler randomly picks up sample packet windows and sends them to the network packet analyser component. The analyser and the preprocessing engine analyse the packets and convert them into a standard XML format. This metadata is sent for processing to the next component, that is, the "rules engine." The rules engine facilitates the XML packet to be checked for anomalies against suspicious activities and predefined site rules. The rules engine should enable the organisation to implement and customise the rules based on the location of the IDS on the network. Then, the packet is sent to the verifier, which checks it against attacks picked from a local signature database. This database is updated through the updater and digester component that shared the signatures among all IDS instances on the network. The updater listens for signature updates daily from the master database, which is connected to the cloud and sends a web service that pushes signature updates to all instances. The updater then picks up these XMLs and their packet payloads and digests them using hashing algorithms and stores them in the local signatures database. Since the signatures are hashed, comparing them in the verifier against new XMLs and network packet payloads becomes easy and quick, thus achieving the "fast Ethernet" speeds that this architecture claims.

The updater and digester component ensure that the

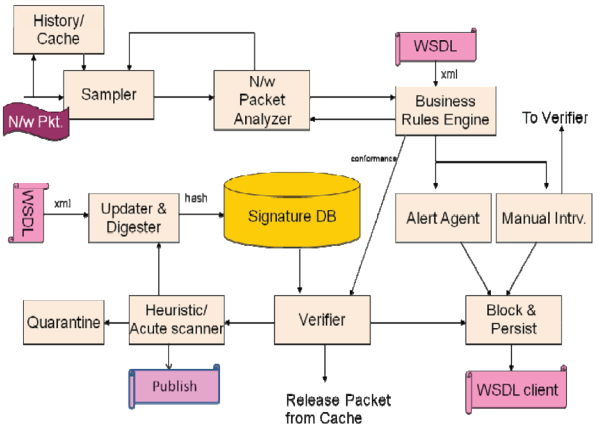

Fig. 3. Multiple site's IDS architecture.

Algorithm 1: Generate and update signature databases

1: Begin
2: N = 0 # number of current signatures
3: Query the MySQL database to retrieve the set of signatures detected, *S*.
4: **for** every signature *f* in *S* do
5: *Freq* = number of occurrences of *f*
6: *LTime* = last detection time of *f*
7: **if** N <= *MaxNum* and *Freq* >= *MinFreq* and *Ltime* >= *ValidTime* **then**
8: Remove the signature from the secondary database
9: Add the signature in multiple IDS
10: N = N+1
11: **end if**
12: **end for**
13: Restart multiple and complementary IDS
14: End

Figure 4- Algorithm for updating signatures database.

local database is updated with unknown attacks and, thus, can prevent them as well, making it a true IDS.

Uddin et al. [7] believed that it is easy to facilitate a dynamic continuous update for small signature databases, compared to complementary large database. They proposed a new multilayer model for signature-based IDSs consisting of multiple small signatures databases (SSDs), which contain the most frequent attack signatures, and a complementary database that contains all the signatures gained from the training process, which are used to update the small databases using mobile agents at a certain time interval. The mobile agents are used to automatically update the SSDs with new attack signatures from the complementary database at regular intervals in order to address the main problem of SIDS,

which is how to deal with the large amount of traffic that makes up each packet to be compared with every signature in the database (traffic flooding on the network) in order to detect the imminent threats. But their model does not have any mechanism for adding, removing, or updating the complementary database.

Uddin et al. [7] gathered information about the threats that may appear during the flow of traffic for a period of time on the targeted network, which is called the training process. They identified the most frequent attacks based on three factors:

1) The minimum number of occurrences of a signature (MinFreq)

2) The age of the alert in the database (ValidTime)

3) The maximum number of signatures that the user would like to keep in all the IDSs.

Uddin et al. [7] further developed an algorithm using mobile agents to create the most frequent signatures databases and updated the small databases at regular intervals.

In this experimental setup, Uddin et al. [7] chose SNORT as the signature-based IDS platform and the MySQL database to log the alerts, which performs on two different hardware platforms: the IDS run on the lower machine while the attacking tool runs on the faster machine because the attacker's resources mostly overwhelms the IDS resources.

The work is performed using a commentary database that contains all attack signatures recognisable during the training process and a set of small databases that contains the most frequent attack signatures during the training process, with a mobile agent for every small database to update with a new attack signature once it is added to the complementary database. But the proposed model does not have any mechanism for adding, removing, or updating the complementary database, which means if new attacks are not added to the complementary database as quickly as they are created, the agents would be useless.

Umar et al. [8] attempted to overcome one of the main challenges of signature-based IDS (SIDS), which is how to control the huge traffic volume so that each packet can be compared with each signature in the known signature database. Therefore, they proposed a new architecture using multiple agents with small databases for each to achieve a high success rate of attacks detected by dynamically updating the agents' signatures databases from the main complementary database at regular intervals, with the aim of reducing the time needed to compare the signatures and automatically update the small databases in the agents. They believed that, without updating the IDS, small changes in the known attacks will not be detected. Their proposed architecture combined the two approaches of multithreading and parallelising IDS, as shown in Fig. 4.

In each agent's architecture, the intrusion detection module takes the packet as input, extracts the signature, and compares it with available signatures in its SSDs. If any match occurs, the incoming packet is considered as an attack, a log record is created of that attack at the main server database, an alarm is generated, and that packet is blocked. If there is no match, the packet is considered to be a normal packet, and it is allowed to pass to the targeted network. The second module is the SSD update module, which updates the agent's SSDs, which depends on the update algorithm based on different parameters, such as most frequent attacks and log record and the age of the signature alert. The updating module is responsible for updating the agent's small databases from the main server database, as shown in Fig. 5.

The experiment was performed by choosing two agents with a specific number of signature rules in its small databases. Snort was used as the IDS platform, and BASE was used as the security analyser. All the alerts were stored in the MYSQL database. An IDS wakeup was used to check the performance of the proposed

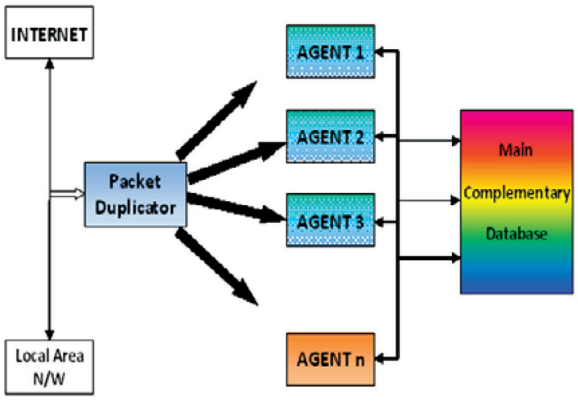

Fig. 5. Proposed model of the IDS.

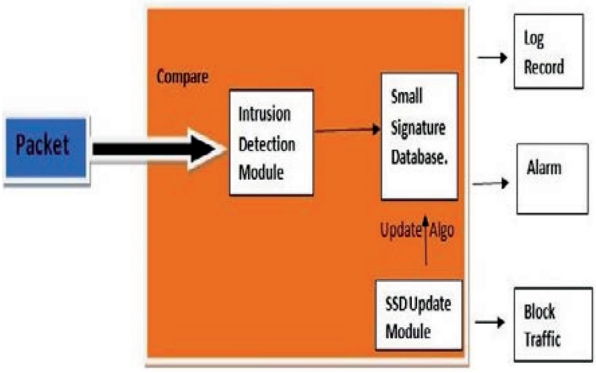

Fig. 6. Structure of the small database's mobile agent.

architecture by generating some common attacks. Then, the results were compared with the traditional technique of using a single large database containing an entire set of rules and sending the same attacks to both methods for processing. The result proved that the traditional technique took twice as long as the study method.

Umar et al.'s model [8] uses multiple agents, each with its own signatures and rules. Once the new packet is received, it is duplicated to every agent, which in turn analyses the packet and compares its signature with the agent's signatures database to decide whether it is normal or not. These small agent's databases are updated automatically from the complementary database at certain intervals. Their model depends on how fast the complementary database is updated, which means if new attacks are not added to the complementary database, the agents will be useless.

In their study, Folorunso et al. [9] agreed that keeping the database of known attack signatures updated and the setting of a suitable threshold level for intrusion detection is the biggest problem faced by SIDS. They proposed (CA-NIDS), which uses three databases to enable the SBS to detect new attacks using a combinatorial algorithm. These three databases are the attack signature database, the new attack database, and the normal traffic database. Folorunso et al. [9] chose the combinatorial algorithm for the analysis engine, and they used a threshold value of 12 for sequence matching and intrusion classifications, for which a match score value less than the threshold value would be classified as intrusion and values greater than or equal to the threshold value would be normal.

As with any IDS, this one requires four basic components: the sensor, the database of attack signatures, the matching engine, and the analysis engine. In their study, Folorunso et al. [9] employed two sensors, one for the matching engine and the other for updating the new attack signature from the IDS provider. The sensor extracts features in the incoming traffic and transfers them to the matching engine, where they are then compared to the extracted traffic with the known attack signatures database. If there is a match, then, it is flagged as unknown traffic and transferred to the analysis engine, which in turn performs a combinatorial analysis between the unknown attack and the normal traffic database by computing similarities between the normal traffic database and the unknown traffic data to classify the traffic data as an attack or normal data. Then, the network administrator takes the appropriate action of whether or not to update the new attack database with the new analysed unknown traffic.

The network administrator is saddled with the task of using the signature from the newly detected attack to update the new attack database. The next algorithm describes this process.

Thus, they use false positive and accuracy metrics to evaluate their CA-NIDS and other known network algorithms, as shown in Table II.

This model consists of three phases, the sensor, the matching engine, and the analysis engine, as is the case with any IDS in which two sensors are applied. The three databases are deployed in the matching engine, and the combinatorial are deployed in the analysis engine. When the matching engine does not detect any threat in the incoming packet, it transfers it to the analysis engine for further analysis by matching it with the normal traffic database to decide the similarity rate between them. If no similarity is detected, the packet will flag it as abnormal and assign the administrator with task of deciding whether to add this attack to the new attack database. This is prone to error because the new attack database is connected to the known signatures database for continuous updating once the new attack database is updated, which means if the administrator makes the wrong decision, it will negatively affect the whole architecture performance.

Almutairi [10], in his 2017 paper, discussed one of the major drawbacks of IDSs, which is that the size of signatures database must be addressed and then updated regularly. He proposed a module consisting of an IDS

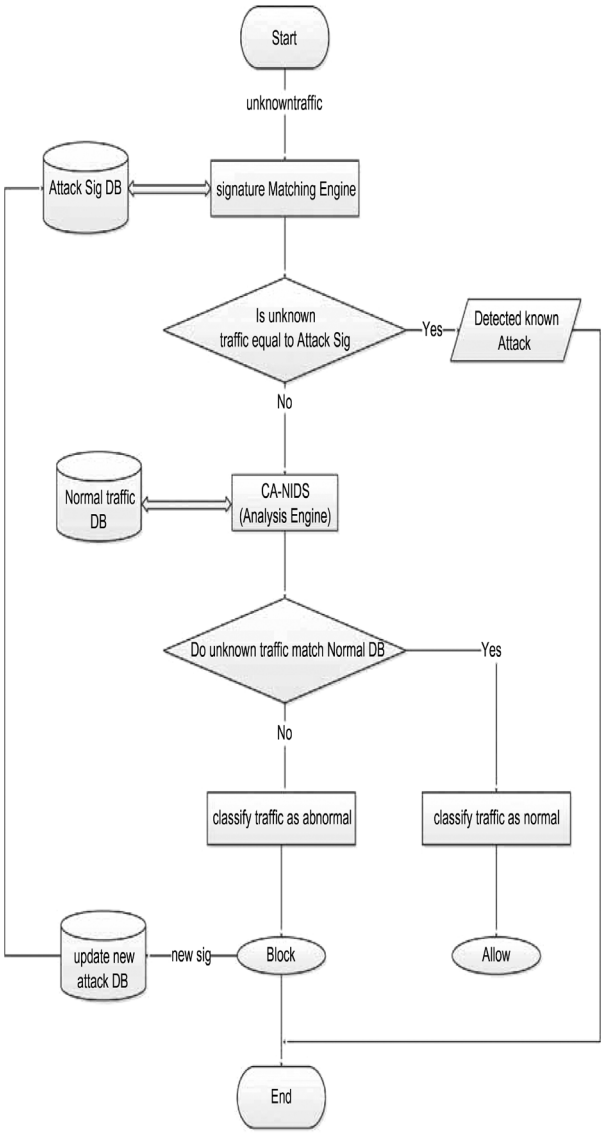

Fig. 7. Flowchart for CA-NIDS.

engine, small signature databases that store the most frequently used signatures, an updating agent, and a complementary database that stores the remaining signatures.

The updating agent was used to update the small signatures databases with the latest signatures and to remove signatures that became infrequent over time using the complementary database.

Almutairi [10] developed an algorithm for updating the small frequency databases from the complementary database to be deployed in the updating engine Fig 9.

The updating agent checks the following three conditions to determine whether or not to add the signature to the small database:

1. frequency value:
   check if it is greater than or equal to the minimum frequency value;
2. last alert time:
   check if it less than or equal to the valid alert time; and
3. the number of signatures in the small signature database:
   check if it is lower than the maximum number allowed in this database.

If these conditions apply, then the signatures that match these conditions will be moved from the complementary database to the small database.

In some cases, some of the nonfrequent signatures must be added to a small signature database for faster detection. So, there should be a process for adding these important nonfrequent signatures. The process of adding such new signatures to the small signature database is done according to these predefined priorities that are decided manually by the administrator, as shown in Table III.

TABLE II
PERFORMANCE RATE OF THE PROPOSED IDS MODEL

Algorithms	Accuracy (%)	False positive (%)
CA-NIDS	96.5	3
K-NN (Eskin, 2002)	91	8
Clustering (Lee, 1999)	93	10
SVM (Eskin, 2002)	91	6
SOM (Kayacik, Zincir-Heywood, 2003)	90.6	7.6
Decision stump+traditional Adaboost (ADST) (Hu et al., 2014)	90.13	2.23
Online GMMs+new Adaboost (AGMM) (Hu et al., 2014)	90.61-91.15	1.17-1.69

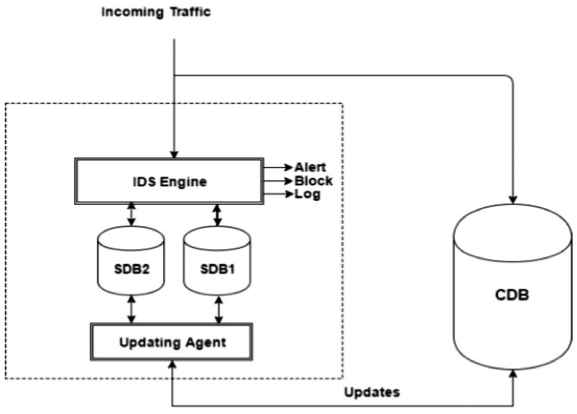

Fig. 8. Structure of proposed model.

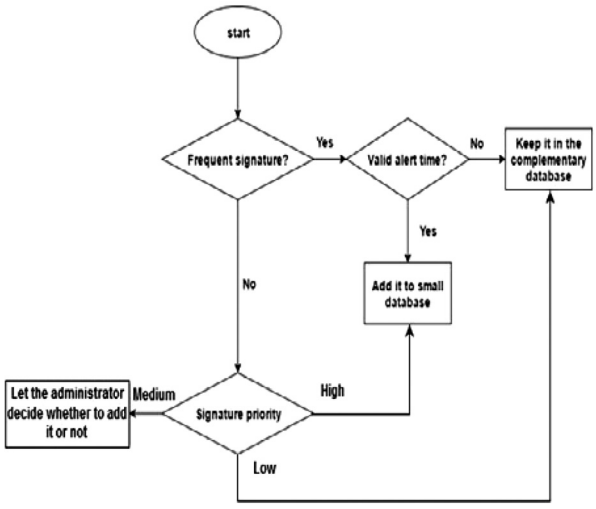

Fig. 9. The order of the updating process.

The paper addressed the large size of the signatures database by distributing the most frequent signatures into smaller signatures databases based on predefined factors and storing the remaining less frequent signatures in the big database, which it used mainly to update the smaller databases with the new attack signatures using a mobile agent at certain intervals. Therefore, when a new attack is received, the IDS will not detect it unless the complementary database updates the agents with that new attack signature. Also, assigning the network administrator the task of adding or removing signatures to or from the small databases is labour-intensive process and prone to errors.

```
Let:
Sig = individual signature.
Priority = signature priority number.
SDB = small signature databasewith most frequent signatures.
CDB = complementary database that store all signatures.
MinFreq = Minimum of number of occurrences of Sig to be considered
frequent.
Ltime = Last detection alert time of Sig.
Freq = Number of occurrences of Sig.
N = Number of signatures in SDB.
ValidTime = Valid alert age of a signature.
MaxSize = Maximum number of signatures in SDB.
# Start from here
# initially SDB is empty
For each Sig in CDB Do
If Freq >= MinFreq && LTime <= ValidTime && N < MaxSize then
       {
       Move Sig from CDB and add it to SDB;
       N++;
       }
Else if Freq >= MinFreq && LTime <= ValidTime && N = MaxSize then
       {
       Set each Sig in SDB to SigOld then
       If Freq >= Freq of SigOld && LTime < LTime of SigOld Do
          {
          Remove SigOld from SDB and store it in CDB and move new
          Sig from CDB to SDB;
          }
       Else do nothing;
       }
Else if Sig Priority = 1 then
{
Move Sig from CDB and add it to SDB
}
Else Do nothing;
```

Fig. 10. Algorithm for updating the small frequency databases from the complementary database.

IV. Conclusion

In this survey paper, we covered various types of intrusion detection models that focus on one of the major challenges of IDSs, which is packet overload, allowing a packet to match with a lower number of signatures rather than matching with a huge signatures database in order to avoid packet dropping in case of network flood. In addition to dividing the large signatures databases into smaller ones, there is a need to automate the updating process that updates the small signatures databases and the complementary database as well through building an intelligent model using machine learning rather than manually updating the complementary database only through the cloud or internet, or by the network administrator.

TABLE III
PREDEFINED PRIORITIES BY NETWORK ADMINISTRATOR

No	Priority	Description	Action
1	High	Risk of this malware is high and it might occur in the network soon and need to be detected immediately	Add it to the small signature database for detection
2	Medium	Risk of this maleare is not high and it might or might not occur in the network	Let administrator decide whether to add this signature to the small signature database or not
3	Low	Risk of this malware is low and it might not occur in the network soon	Keep this malware signature in the complementary database and not add it to the small database

REFERENCES

[1] P. Bunel, “An introduction to Intrusion Detection Systems,” in *SANS Conf*., London, UK, June 2004.

[2] U. Bashir and M. Chachoo, "Intrusion detection and prevention system: Challenges & opportunities," in *2014 Int. Conf. on Comput. Sustain. Glob. Develop. (INDIACom)*, New Delhi, pp. 806-809. doi: 10.1109/IndiaCom.2014.6828073.

[3] K. R. Rao, A. Pal, and M. Patra, “A service oriented architectural design for building intrusion detection systems,” *Int. J. Recent Trends Eng*., vol. 1, no. 2, May 2009.

[4] A. Sharma, A. K. Pujari, and K. K. Paliwal, "Intrusion detection using text processing techniques with a kernel-based similarity measure," *Comput. Secur.*, vol. 26, no. 7-8, pp. 488–495, Dec. 2007, doi: 10.1016/j.cose.2007.10.003.

[5] C. Xenakis, C. Panos, and I. Stavrakakis, “A comparative evaluation of intrusion detection architectures for mobile ad hoc networks,” *Comput. Secur.*, vol. 30, no. 1, pp. 63-80, Jan. 2011, doi: 10.1016/j.cose.2010.10.008.

[6] M. Salour and X. Su, "Dynamic Two-Layer Signature-Based IDS with Unequal Databases," in *Fourth Int. Conf. Inf. Technol. (ITNG'07)*, Las Vegas, NV, 2007, pp. 77-82, doi: 10.1109/ITNG.2007.80.

[7] M. Uddin, A. A. Rahman, N. Uddin, J. Memon, R. A. Alsaqour, and S. Kazi, “Signature-based multilayer distributed intrusion detection system using mobile agents,” *Int. J. Netw. Secur.*, vol. 15, no. 2, pp. 97–105, Mar. 2013.

[8] H. G. A. Umar, C. Li, and Z. Ahmad, “Parallel Component Agent Architecture to Improve the Efficiency of Signature Based NIDS,” *J. Adv. Comput. Netw.*, vol. 2, no. 4, pp. 269-273, Dec. 2014, doi: 10.7763/JACN.2014.V2.124.

[9] O. Folorunso, F. E. Ayo, and Y. Babalola, “Ca-NIDS: A network intrusion detection system using combinatorial algorithm approach,” *J. Inf. Priv. Secur.*, vol. 12, no. 4, pp. 181–196, Dec. 2016.

[10] A. H. Almutairi and N. T. Abdelmajeed, "Innovative signature based intrusion detection system: Parallel processing and minimized database," in *2017 Int. Conf. Frontiers Adv. in Data Sci. (FADS)*, Xi'an, 2017, pp. 114-119, doi: 10.1109/FADS.2017.8253208.