ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
https://doi.org/10.17559/TV-20210325173601
Original scientific paper

Application of PSVR-DNS Algorithm for Attacker Detection and Isolation

Denis PEJIĆ*, Višnja KRIŽANOVIĆ, Drago ŽAGAR

Abstract: The DNS (Domain Name System) is used to map and convert human-friendly domain names to the numeric IP (Internet Protocol) addresses. As with the operation of any communication system, there are some security risks associated with the operation of DNS. Actions targeting the availability or stability of a network's DNS service are considered DNS attack. For example, a high volume of traffic and a large number of requests coming to DNS servers are part of a type of DoS (Denial of Service) attack that uses DNS for amplification. Although most DNS servers are open source, some commercial protective DNS services are available for network traffic control, filtering and automatic blocking of requests to undesirable, dangerous or malicious internet domains, but the price of such services is high. In this paper, a new PSVR-DNS (Probability Support Vector Regression-Domain Name System) algorithm is proposed for the purpose of detecting and isolating attackers who pose a threat to an uninterrupted work of the DNS servers. The main focus is on the prevention of the DNS cache poisoning. The collected results showed that the proposed PSVR-DNS algorithm achieves better performance related to faster detection and isolation of attacks compared to some existing algorithms.

Keywords: Bayesian probability; DoS attack; PSVR-DNS algorithm; SDN

1 INTRODUCTION

In this paper, the main focus is on the prevention of the DNS (Domain Name System) cache poisoning. For this reason, this paper proposes an algorithm that will successfully isolate the attacker to prevent his malicious actions. The attacker's goal is to forward a false message that the client will trust and allow its inclusion in his cache. The attacker pretends to be a real DNS server and forwards the fake DNS responses to the client or the other DNS server since they store the first received DNS response. In this scenario, the incorrect DNS response information is stored in the DNS cache [1]. The mentioned attack is called the DNS cache poisoning.

After poisoning the DNS server cache, the DNS server starts spreading fake data to every client computer that sends its DNS query [2]. In the case that the attacker caches incorrect domain IP address information in the DNS server cache, all traffic on that domain is redirected to the computer monitored by the attacker. In this way, the attacker carries out other attacks: eavesdrops the traffic, exchanges arbitrary traffic, compromises sensitive data, spreads malware, and so on. Depending on the fake DNS response, the attacker can perform the following listed types of attacks [1].

1.) Finding other people's sensitive information: In this type of attack, a fake response instead of a real IP address consists of the IP address of a server controlled by the attacker's website (the IP address of the real website has changed with its fake IP address). Bank's website phishing attack can be presented as an example when the attacker redirects all bank website traffic to his fake website, and then collects information about customer passwords that appear on its fake website, and passwords stolen from customers can be misused for unauthorized access to a bank's accounts.

2.) The man in the middle attack: The attacker interferes in communication between two users trying to communicate with each other, and when poisoning the cache, the attacker enters arbitrary data into the cache of the name server. In this way, the name server responds to the user's queries using the values that the attacker entered in his cache, and for this reason the user can be redirected to the wrong server. When a sender forwards packets to a recipient, in reality it forwards them to the attacker, the attacker can intercept, replace or read the packets and then send them to the recipient. When the recipient forwards the packets back, the attacker can also intercept, replace, or read the packets, and then return them back to the sender. In this way, the sender and receiver do not know that the attacker is eavesdropping on them, and the attacker can reveal secret information or control communication by changing an integral part of the packet. Many researchers have tried to develop more efficient models to troubleshoot the DNS cache poisoning and DNS rearrangement attacks on wired or wireless LAN's (Local Area Networks) [3]. This paper proposes an algorithm that precisely prevents the cache poisoning attacks from the DNS attacks. An algorithm based on Bayesian probability and SVR (Support Vector Regression) should detect and prevent an attack on the DNS by analyzing the collected information from the current and previous sessions, and take into account the scenario of equipment failure or technical error. The probability and prediction algorithm is mainly based on the Bayesian probability and the machine learning model SVR [5], to predict the malicious actions of the attacker host (The Internet of things (IoT) node) using the prior knowledge. When obtaining the prediction results, the specific host (IoT node) is analyzed using an attack detection tool. By blocking communication with other hosts on the network, the SDN (Software Defined Network) manager performs the process of isolating the attacking host (IoT node) from the network. Also, the SDN manager during the isolation of the attacker host simultaneously creates new transmission paths at the time of recovery of the attacked hosts (IoT nodes). The rest of the article is organized as follows. In Section 2 the latest research on methods of protection against DNS attacks is described. In Section 3 the proposed PSVR-DNS mechanism for the detailed prevention of DNS poisoning attacks is presented. In section 4, the proposed PSVR-DNS algorithm is compared with the existing DNS-injector-Detector and a simple tool to detect algorithms. In section 5 the main conclusions are drawn.

2 RELATED WORKS

The DNS cache poisoning attack poses a serious security threat today. There are many solutions when it comes to preventing DNS cache poisoning attacks, such as DNSSEC (DNS Security Extensions). However, the proposed DNSSEC solution is not widely used due to its poor performance. This section describes new research aimed at protecting against poisoning and attacks on the DNS cache. In [6], Jin and co-workers proposed an advanced detection method against DNS cache attacks using machine learning methods. In order to identify the DNS response packets used in the event of the cache poisoning attack, especially in the case of compromised authoritative DNS servers, they have proposed adding of features isolated based on the standard DNS protocols, as well as heuristic aspects like "timerelated features", "GeoIP (Geolocation Internet Protocol)-related features", and "cached DNS trigger data", etc. In [7], Ma et al., proposed an intelligent defensive solution, where the process of attack and defense is modelled as a static stochastic scenario with incomplete information in limited conditions of rationality. In [8], Wu et al., proposed an additional method for detecting the domain system cache poisoning attacks. A DNS cache server is used to keep a record of the last queries of a domain name. The cache server directly forwards the response packets to the client if the domain name for which the query is requested is stored in it, otherwise the cache server forwards the query request to higher-level servers for subsequent iterative queries. In the case of non-stationarity, the entropy sequence is first modelled by the state space equation, and then the Kalman filter is used to detect the attack [8]. In [9], Hmood and co-workers [9] have proposed a new protection technique called the Adaptive-Cache of DNS (ACDNS). The proposed solution relies on a caching mechanism to prevent these types of attacks.

ACDNS is implemented in a way that is compatible with current DNS standards and is fully compliant with basic protocol processes and the infrastructure presented in [9]. The paper shows that the proposed method partially protects against DNS cache poisoning attacks because it is sensitive to other types of attacks, such as the DNS Amplification attack, which is a form of distributed denial of service attack, where attackers use DNS servers to flood the main target with multiple DNS response traffic. In [10] Wu et al., proposed methods for detecting and recognizing attacks based on data merging. They found that entropy Internet Protocol (IP) query addresses for all cache servers are approximately stationary and statistically independent in common cases. In the case of a distributed attack, the paper showed that the entropy correlation among all cache servers could increase dramatically. Based on this feature, principal component analysis is applied to the design of detection and identification methods. The paper shows that the attack is true when the maximum eigenvalue of the normalized entropy matrix exceeds the threshold, and the attacked servers are recognized as the main load vector [10]. In [11], a solution is presented whose primary goal is to construct an abnormal return trip time profile from the moment the response is sent to the opponent, and then to look for constructive profile anomalies. If the DNS server cache is poisoned, the attacker must know the source port and Transaction Identifier (TID) of the request. If the attacker cannot capture the request, it cannot recognize the transaction ID. So he has to guess the correct value, and that usually means that it must try sending multiple responses with different TIDs. In [12], the researchers proposed the DNS-IDS (Intrusion Detection System for the DNS protocol) system that operates in two phases, the training phase and the operational phase. In the training phase, the normal behavior of the DNS protocol, as a machine with finite states, is modeled. The normal time statistics are derived related to the way in which normal DNS traffic is transmitted using the proposed machine learning model, and then stored in a database. In the operational phase, anomaly metrics are used to detect the DNS attacks, both known and new attacks. Based on the proposed solutions listed in the literature, it is considered that the problem of detection of attacking hosts (IoT nodes) can be considered and solved using a new proposed PSVR-DNS algorithm presented in this research paper.

3 APPLICATION OF PSVR-DNS ALGORITHM IN DETECTION OF DNS CACHE MEMORY POISONING

This chapter will describe the application of SDN managers and the application of machine learning to detect enhanced DNS attacks. When it comes to security, defining and executing a security configuration for applications, frameworks, Web servers, database servers, and so on is required. For this reason, adequate protection against the DNS spoofing attacks is required. This research paper will describe the theoretical aspects of an algorithm based on machine learning and probability assessment that effectively protects against DNS spoofing attacks.

3.1 Architecture of the Proposed PSVR-DNS Algorithm

This chapter describes the architecture of an algorithm based on probability assessment, machine learning, and abnormal packet detection in the event of a DNS attack. The proposed PSVR-DNS algorithm, with regard to the research conducted in the paper [4], has a table which comprises the lists of abnormal packets whose application more effectively protects the user from malicious attacks. In Fig. 1, the proposed architecture is presented. It consists of the three main processes: a prediction probability algorithm, an abnormal packet detection, and an SDN (Software Defined Network) based attack check. The architecture of the proposed algorithm shown in Fig. 1 is based on the process of monitoring DNS cache, and is modeled based on the basic configuration of the agent-server in networks with the DNS routing. The algorithm architecture consists of a server and an agent, with separate management for each.

The server includes: data display, client control, detection of abnormal packets, attack protection and server control. Data display performs host domain monitoring, packet monitoring, duplicate checking, a list of blocked attack nodes. The agent manager maintains all agent information such as agent status, network control, network configurations, used to prevent security threats. The abnormal packet detector updates the table comprising abnormal packet list gathered based on the results. The packet is detected as abnormal in the following cases:

1.) No IP address-domain name pair is found in the DNS table.
2.) An IP address domain name pair was found, but the timestamp is not accurate (timestamp represents the current time of the event recorded by the computer).
3.) A suspicious pair of IP address-domain name was found. Server Protection performs the attacker decision making process, generates a warning message when an abnormal packet detects it, and initiates the removal of the host (IoT node) from the network.

The server manager collects information about agents, hosts, abnormal packet list tables, server security modules, and manages the network, while also representing the module through which the server communicates with the agent. The agent includes detection modules, managers, and security modules. The agent detection module is the main module in this research where the main attack detection procedure is performed. The detection module includes a probability prediction-based algorithm, additional attack checking based on routing traces, and also includes a timestamp field to avoid DDoS (Distributed Denial of Service) attacks. The host receives DNS request and response packets, and at the same time, a timestamp is generated for the DNS request and response packets. The algorithm generates a warning message whenever an expired time stamp or packet abnormality is received in the packet. For this reason, the table comprising abnormal packet list is created to provide resistance to DoS/DDoS attack. The algorithm also processes the message using the opcode test, consistency test, and timestamp expiration to generate a warning message if the packet fails the test.

The detection module initially compares a pair of IP domain names using probability prediction on the DNS cache, global DNS cache, processes the DNS request by performing multilayer consistency testing, abnormal packet detection by validation, or timestamp expiration, if suspicious addresses are received in the DNS table. In case the packet is suspicious, it is added to the table with the list of abnormal packets in order to avoid the same attacker endangering the host. In doing so, the table with the list of abnormal packets is managed by a counter that has the task of showing how many times the same pair of IP domain names has been entered. Furthermore, this information is sent to the agent manager who confirms the occurrence of the attack and notifies the agent protection module to change the DNS type. After that, the recovery process is initiated. The agent manager also performs data collection, command forwarding, status checking, and communication with the server manager.

3.2 Attack Prediction Relations

If the set $S = \{S_y, 1 \le y \le n\}$ denotes a host (IoT node) with n host characteristics, the host sets can comprise features of an attacker host (malicious node) or a normal host [4]. The S_y tag is used when determining the host configuration, that is, it uses the network information of each host when detecting an attack. Here n denotes host's features, i.e., information whether it is offensive or not offensive host. The probability that the host is an attacker is denoted as $V\left(x_j^{attacker}\right)$, while the probability that the host is not an attacker is denoted as $V\left(x_j\right)$. After the SDN manager identifies the host feature, the probability is calculated using the following Eq. (1) and Eq. (2) [13]:

$$V\left(x_j \mid S_y\right) = \frac{V\left(x_j\right)V\left(S_y \mid x_j\right)}{V\left(x_j^{attacker}\right)V\left(S_y \mid x_j^{attacker}\right) + V\left(x_j\right)V\left(S_y \mid x_j\right)} \quad (1)$$

$$V\left(x_j^{attacker} \mid S_y\right) = \frac{V\left(x_j^{attacker}\right)V\left(S_y \mid x_j^{attacker}\right)}{V\left(x_j^{attacker}\right)V\left(S_y \mid x_j^{attacker}\right) + V\left(x_j\right)V\left(S_y \mid x_j\right)} \quad (2)$$

In doing so, the SDN manager has the responsibility to classify the host (IoT node) as an attacker depending on the probability value. A higher probability value classifies the host (IoT node) as an attacker. It should certainly be noted that erroneous detection is possible if one is to rely solely on that probability value. In addressing this shortcoming, the following features must be monitored, and a list of features must be maintained. The list of features is constructed from $ZN = \{ZN_i, 1 \le i \le n + q\}$ with $n + q$ features, where features can be normal or offensive. By applying simplified posterior probability in an iterative manner, the host can be identified as an attacker (malicious) or a normal host (IoT node). A machine learning model is used for the iterative procedure and the posterior probability is written as [4, 13]:

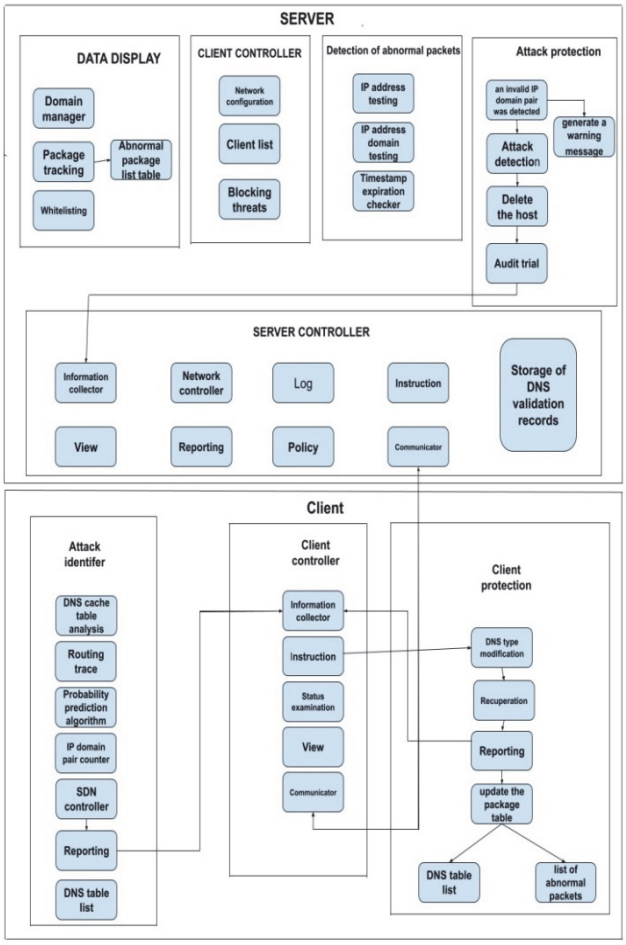

Figure 1 PSVR-DNS algorithm architecture

$$V\left(ZN \mid S_y\right) = \frac{V\left(ZN\right)V\left(S_y \mid ZN\right)}{V\left(S_y\right)} \quad (3)$$

It is necessary to show the features of the support vector regression (SVR) hyperparameter for the formulation. When calculating the probability, the distribution of these properties $V(ZN|S_y)$ is needed. For the estimate $V(ZN|S_y)$ a flat distribution is assumed, and by applying the Bayesian probability to the SVR $V(S_y|ZN)$ with the help of the Taylor expansion, the integral is written as follows [4]:

$$V\left(S_y \mid ZN\right) = \int_0^{n+q} V\left(S_y \mid f, ZN\right) V\left(f, ZN\right) \mathrm{d}f \tag{4}$$

With regard to the previous statements, the probability $V(S_y|ZN)$ can be written as:

$$V\left(S_y \mid ZN\right) = L_f^{-1} L_{ZN}^{-1} \int_0^{n+q} \exp\left(-ZN\left(f\right)\right) \mathrm{d}f \tag{5}$$

Here, L denotes the noise function associated with the transmitted frames. In the case in which the Taylor expansion for $ZN(f)$ is considered, the probability becomes extended as [4]:

$$\begin{aligned} &V\left(S_y \mid ZN_i\right) = \\ &= L_f^{-1} L_{ZN}^{-1} \int_0^{n+q} \exp\left(-ZN_i\left(f\right), \ldots, ZN_{n+q}\left(f\right)\right) df \end{aligned} \tag{6}$$

Based on these probabilities obtained from the Bayesian probability and SVR hyperparameters, the probability of predicting an attack can be formed as follows [4]:

$$\begin{aligned} V\left(x_j \mid ZN_i..ZN_{n+q}\right) + V\left(x_j^{attacker} \mid ZN_i, ..., ZN_{n+q}\right) = \frac{V\left(x_j\right) V\left(ZN_i, ..., ZN_{n+q} \mid x_j\right)}{V\left(x_j\right) V\left(ZN_i, ..., ZN_{n+q} \mid x_j\right) + V\left(x_j^{attacker}\right) V\left(ZN_i, ..., ZN_{n+q} \mid x_j^{attacker}\right)} + \\ + \frac{V\left(x_j^{attacker}\right) V\left(ZN_i, ..., ZN_{n+q} \mid x_j^{attacker}\right)}{V\left(x_j\right) V\left(ZN_i, ..., ZN_{n+q} \mid x_j\right) + V\left(x_j^{attacker}\right) V\left(ZN_i, ..., ZN_{n+q} \mid x_j^{attacker}\right)} \end{aligned} \tag{7}$$

This Eq. (7) can be simplified when designing algorithm based on Bayesian probability and SVR final iterative probability prediction. By retaining the conditional probability based on the list of features and the configuration of the host [4]:

$$\begin{aligned} &V\left(x_j^i \mid ZN_i\left(S_y\right)\right) = \\ &= \frac{V\left(x_j^i\right) V\left(ZN_i\left(S_y\right) \mid x_j^i\right)}{V\left(x_j^i\right) V\left(ZN_i\left(S_y\right) \mid x_j^i\right) + V\left(x_j^{attacker(i)}\right) V\left(ZN_i\left(S_y\right) \mid x_j^{attacker(i)}\right)} \end{aligned} \tag{8}$$

If applying constraints to Eq. (8), it helps in determining if a host is an attacker or not. Based on the results obtained by the constraint on the Eq. (8), the following three cases are concluded for iterative hosts [4]:

1.) If $V\left(x_j \mid ZN_i\left(S_y\right)\right)$ increases with the appearance of the host features for the formula from the Eq. (8) $V\left(ZN_i\left(S_y\right) \mid x_j^i\right) > V\left(ZN_i\left(S_y\right) \mid x_j^{attacker(i)}\right)$, then the probability of being an attacker is higher than that of not being an attacker.

2.) If the value $V\left(x_j \mid ZN_i\left(S_y\right)\right)$ is close to 1 when a new host feature appears, then $V\left(ZN_i\left(S_y\right) \mid x_j^i\right)$ will be close to 1 or the host will be the attacker.

3.) If $V\left(x_j \mid ZN_i\left(S_y\right)\right)$ is close to 0, when the host feature for the Eq. (8) from the relation $V\left(ZN_i\left(S_y\right) \mid x_j^i\right)$ appears, it most likely becomes zero or close to zero, indicating that the host is not an attacker.

3.3 Description of the PSVR-DNS Algorithm

The probability and prediction algorithm is mainly based on usage of Bayesian probability and machine learning model (Support Vector Regression), to predict the characteristics of the malicious actions of the attacker host (malicious IoT node) using the prior knowledge. When obtaining the prediction results, the specific host is analyzed using an attack detection tool, and the recovery tool isolates it from the network. The host isolation procedure allows the SDN manager to isolate a host (IoT node) from the network to block communication with other hosts on the network until the host is free again. By isolating the host (IoT node), possible attacks are prevented in such a way as to prevent lateral movement through other hosts. The SDN manager performs the process of isolating the infected hosts and, at the same time, enables new paths for transmissions at the moment when the recovery process of the infected hosts begins. The PSVR-DNS algorithm prevents an attacker from creating a special case where isolating a host from its own IP address results in isolated hosts successfully communicating with the rest of the network, even when they are successfully blocked from the network. The SDN manager is used to collect all DNS packets that help in obtaining a global database of all client sessions (agents) that are applied for the purpose of attack detection. The attack detection procedure using a probability prediction-based algorithm and threshold-based attack verification is shown in Fig. 2. The PSVR-DNS algorithm continuously monitors local DNS records to prevent unexpected modifications by malicious attacks and localized human error. In this way, the PSVR-DNS algorithm achieves, by constantly monitoring DNS records, the necessary security in terms of directing traffic to websites, electronic communication, and more. Using a machine learning model SVR and a Bayesian probability, it predicts the probability for a specific server.

Then an SDN manager is applied to record the number of malicious attackers. The attacker's goal is to poison the DNS server's cache. The SDN manager notes possible attacks, such as DDoS, where an attacker deploys many compromised hosts (IoT nodes) as the source of executing attack traffic. The SDN controller detects the attack based on flows containing fake IP addresses sent from one or more sources by applying machine learning and Bayesian probabilities. In such a way, the SDN controller checks its destination address for each incoming packet, and in case the destination address is unknown, it forwards the packet to the table comprising abnormal packet list where further checks are performed. If an abnormality of the package is observed, then an additional check is carried out by applying a probability threshold that is adjusted depending on the type of attack. If the probability of a suspicious host (IoT node) is higher than a dynamically defined threshold, the PSVR-DNS algorithm isolates it from the network. After discovering the features of the attack on the server, the probability is refreshed. Some of the DNS server features are: Authoritative and Recursive DNS server, Windows Remote Management, Server Core and DNS server engine. The attacks can harm the user by redirecting their host to the wrong web page. The PSVR-DNS algorithm, after detecting server features, performs filtering of received responses that do not match the sent request. After that, an authorization check is performed for all incoming DNS responses, and the records for which the DNS server does not have authorization are not taken into account. It also ignores DNS responses that do not come from the IP address to which the DNS request was forwarded.

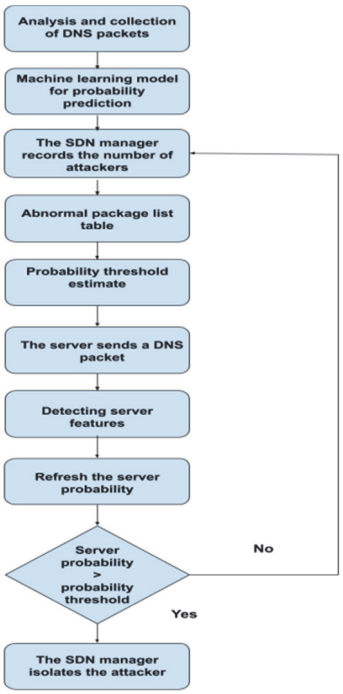

Figure 2 PSVR-DNS algorithm

The idea of the algorithm is to introduce some kind of threshold that plays an important role in detecting and isolating attackers. The threshold value must be dynamically adjusted in such a way as to avoid wrong detection. With the PSVR-DNS algorithm, the threshold value is adjusted depending on the type of attack. For this reason, the following parameters are taken into account for the threshold value: the number of current flows in the router, the number of requests and incoming traffic that generate the IP address, as well as the number of current flows that generate IP address. In this way, if the calculated probability of the host is greater than the threshold, which is adjusted depending on the type of attack, an attack is detected and isolated. The PSVR-DNS algorithm deletes the route to the infected host (malicious IoT node), and its routing table as well.

Likewise, the PSVR-DNS algorithm blocks the communication of the infected host (malicious IoT node) with other hosts on the network until the moment of its isolation from the network. During the process of isolating infected hosts (malicious IoT nodes), the SDN manager enables new transmission paths. The flowchart also introduces some kind of table with the list of abnormal packets, where the algorithm processes the DNS response by checking the operating codes, detects the ”abnormal” packet by timestamp validation, and so on.

4 PERFORMANCE EVALUATION

4.1 Simulation Environment

In the experiment, the results were obtained using an Intel® Core® i7-8550U 1.80 GHz CPU with 8 GB of RAM. In order to perform this experiment, the latest version of VirtualBox software is installed for the purpose of installing the Linux operating system within the Windows environment. After installing the Linux operating system, a Mininet network emulator was installed on the VirtualBox. Mininet is a network emulator that creates a network of virtual hosts (IoT nodes), switches, managers, and connections. Mininet hosts (IoT nodes) run standard Linux network software, and its switches support OpenFlow for highly flexible custom routing and software-defined networking. In this research paper, the proposed PSVR-DNS algorithm is compared with the existing algorithms the DNS-injector-Detector, and a simple tool to detect.

4.2 Simulation Results

In Fig. 3, the manager used in this paper is presented, the SDN manager with its installed components. The SDN manager is connected to the OpenFlow switch (s1).

The topology consists of 31 hosts (IoT nodes) connected to 5 switches. In a network topology, the DNS server represents the main victim of the attack, and thus all normal and compromitted traffic on the switch (s1) passes through the (s1) switch. Switch flows (s1) are continuously monitored in order to obtain the necessary parameters for machine learning. The specified switch parameters (s1) are monitored every 3 seconds using a bash script whose task is to read the flow in the previous 3 seconds, find relevant features from streams by calling the python script processing, and save their value in the file.

Using this script, a training data set is created.

Namely,in the proposed network topology, the normal traffic flows and malicious traffic flows are present. By applying the hping3 command from the proposed network topology, data on normal and malicious traffic flows are obtained. For instance, the attacker 6 pings a DNS server

using a bash script. The specified script provides a random time for each host to ping the DNS server or a random number of packets containing a random number of bytes which should be sent to the DNS server. Normal traffic sends one packet per second, and malicious traffic sends multiple packets per second.

The traffic representing attack is generated by sending many packets (15 packets per second) at high rate. They comprise fake IP addresses sent to the DNS server for the purpose of carrying out the attack. In this research work, an enhanced DNS attack is applied which is a popular form of DDoS attack, i.e. an attack that is performed with multiple computers. The DDoS attack is performed using a botnet. Botnets are networks of computers infected with a Trojan horse or worm that, after sending a command, forward a numerous number of requests to an IP address, so it is very difficult to detect the perpetrator because he may be in a foreign country. The biggest problem with DDoS attacks is that they are easier to be implemented than it is to be defended against them. An attacker simply "employs" computers that have previously been infected with a particular Trojan horse or worm to attack a particular web server. In this way, the algorithm, using machine learning, predicts the characteristics of malicious actions of attackers by applying the learned procedures of performing different types of attacks. The SDN manager records the total number of potential attackers using probability and machine learning. Moreover, the SDN manager examines whether the probability of a potential detected attacker is greater than the defined threshold when the SDN manager isolates the attacker from the network. The paper presents the false DNS detection representation with the proposed PSVR-DNS algorithm, and its comparison with the detection using existing DNS-injector-Detector algorithms and a simple tool to detect. Detection algorithms are compared in terms of detection time, detection error, packet drop rate, attacker isolation and recovery time, and attacker isolation and recovery error.

Detection time represents the time required by the proposed PSVR-DNS algorithm to analyze and detect the presence of an attacker in the network. The graph shown in Fig. 4 represents the detection time required by the proposed PSVR-DNS algorithm compared to existing DNS attack detection algorithms. It can be concluded from Fig. 4 that the proposed PSVR-DNS algorithm requires less time to detect DNS attacks compared to existing detection algorithms, DNS-injector-Detector and a simple tool to detect. It can be concluded that applying a local DNS cache table and a global list of DNS cache reduces the detection time in the proposed PSVR-DNS algorithm. A comparison of algorithms by attack detection error is shown in Fig. 5. Due to the increase in the number of hosts (IoT nodes), the traffic in the network increases, and the detection modules vary when determining normal and attack features on the basis of the DNS packet.

However, the proposed PSVR-DNS algorithm has the smallest error in detecting attackers compared to existing algorithms, the DNS-injector-Detector and a simple tool to detect. Fig. 6 presents a comparison of detection algorithms when it comes to packet drop rate. In this case, the packet drop rate directly affects the number of DNS packets of the host, as well as the congestion of network traffic. When it comes to increase in the number of hosts in the network, the number of DNS request packets and DNS response packets also increases, and this causes congestion and traffic delays, which also serves to packet loss and destructive attacks. It can be seen from Fig. 6

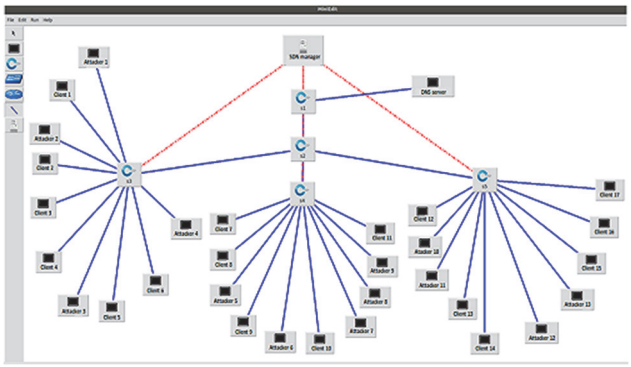

Figure 3 Network topology

That the PSVR-DNS algorithm has the lowest packet drop rate compared to other algorithms, the DNS-injector-Detector and a simple tool to detect. This is, at the same time, the cause of previous attacker detection based on continuous communication between the host and server.

The PSVR-DNS algorithm detected attackers with greater accuracy and in a shorter time interval. The SDN controller used in the PSVR-DNS algorithm effectively isolated the infected hosts (malicious IoT nodes) while creating new transmission paths, including the attacker host recovery procedure. The proposed PSVR-DNS algorithm, shown in Fig. 7, isolated the attackers faster than the existing DNS-injector-Detector and a simple tool in the case of 30 hosts (IoT nodes). Likewise, the proposed PSVR-DNS algorithm, presented in Fig. 8, provided the lowest error in isolating the attacker and recovering in the case of 30 hosts. It can be seen from Tab. 1 that the machine learning based PSVR-DNS algorithm achieves higher detection accuracy, lower false positive and false negative detection rate when identifying attackers if the threshold value is lower than 0,5. It can also be concluded that when the threshold value increases, the PSVR-DNS algorithm distinguishes a normal user from an attacker with more difficulty.

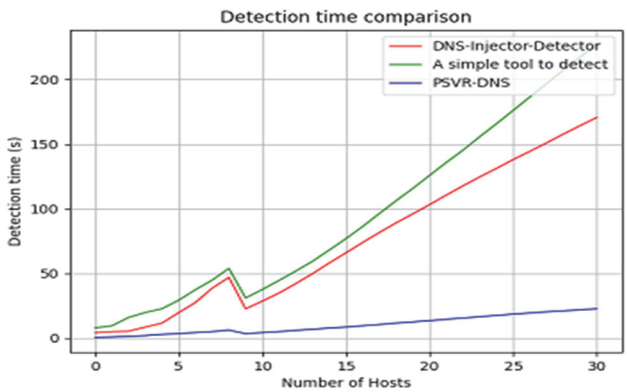

Figure 4 Detection time comparison

Tab. 2 shows a record of detected requests and responses using the SDN manager. Also, it can be seen from the table that, in the case of an attack, the number of possible potential attackers increased sharply compared to

the case when there was no attack. In case when there is not attack, the proposed solution in some situations mistakenly detected the client as an attacker. In the case of an attack, the number of DNS responses is significantly higher than the DNS requests, because a DNS Amplification attack takes place.

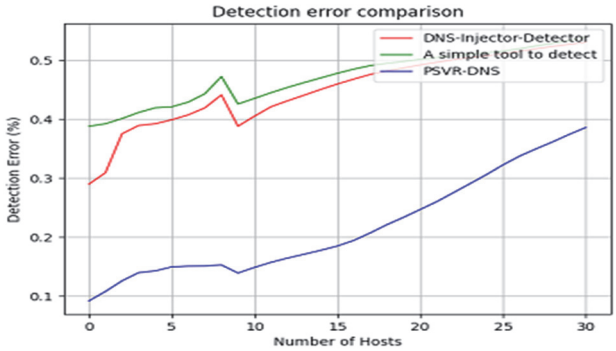

Figure 5 Detection error comparison

A DNS Amplification attack is based on a single DNS request generating multiple DNS responses. Using a forged address sends a DNS query to the DNS server. Furthermore, the DNS server forwards the DNS response to the attacked client. In this way, the attacker sends a DNS query to the server many times in order for the DNS server to forward multiple fake DNS responses in the attacked client's network. This causes a very high exploitation of resources for an attacked client, as well as a loss of access to a particular service. The proposed solution cannot fully detect the attacker, and it cannot prevent false DNS responses if the attacker reaches an attacked client at the beginning of the attack, but when it detects the attacker, the attacker is isolated from the network. Tab. 2 shows that the SDN manager recorded many attackers when it came to an DNS Amplification attack. However, the SDN manager in some cases recorded a higher number of attackers than the total number of attackers, which is 13, because the threshold did not adjust to the total number of identified attacks in a given period of time. By applying the SDN manager, the proposed solution can detect and successfully isolate the attacker from the network. Fig. 9 shows network traffic during a specific attack period. Ten malicious computers were used to generate offensive traffic, and for that reason, the amplitude on the Y - axis is much larger.

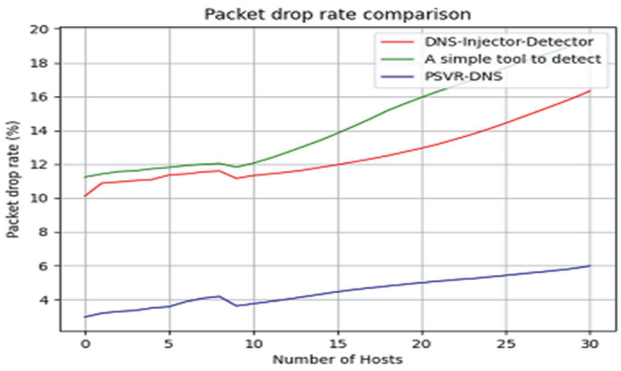

Figure 6 Packet drop rate comparison

Fig. 9 shows the application of the PSVR-DNS algorithm in the event of an attack. The figure shows the beginning and end of the attack in the case when the algorithm is turned off (prevented or disabled), and the case when the algorithm is enabled and detects and isolates the attack based on the defined treshold.

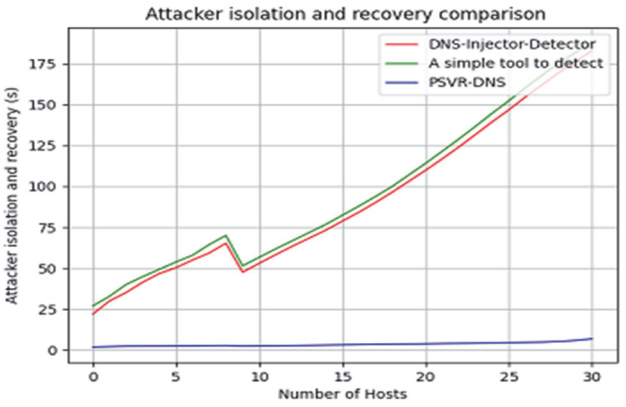

Figure 7 Attacker isolation and recovery comparison

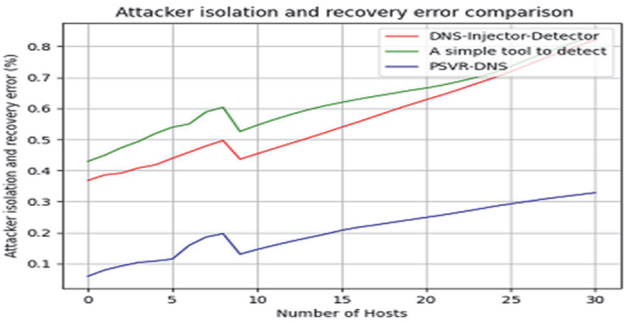

Figure 8 Attacker isolation and recovery error comparison

Table 1 Application of a defined threshold for the purpose of isolating an attacker

Threshold	Detection rate	False positive detection rate	False negative detection rate
less than 0,5	97,29	0,21	2,5
equal to 0,5	81,57	0,85	17,58
greater than 0,5	72,11	2,19	25,7

Table 2 Display of PSVR-DNS algorithm detection

Time	Requests	Replies	Detected Attackers	Description
19:06	6041	5860	12	It's not an attack
19:08	6392	5921	4	It's not an attack
19:09	5899	5729	9	It's not an attack
19:10	1372	2672	18	Attack
19:11	462	2423	7	Attack
19:12	6031	5872	15	It's not an attack
19:30	6693	6282	3	It's not an attack
19:47	382	1099	11	Attack
19:51	599	1894	14	Attack
19:55	1283	2501	6	Attack
20:06	5483	5217	13	It's not an attack
20:19	5302	5021	2	It's not an attack

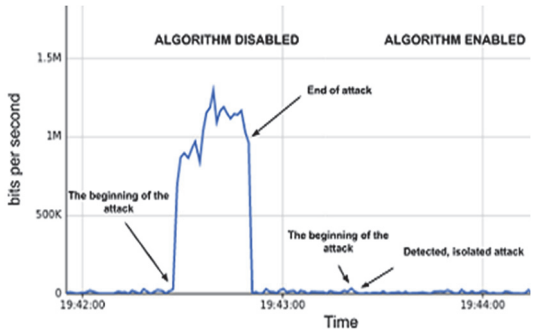

Figure 9 Behavior of the PSVR-DNS algorithm in the event of an attack

5 CONCLUSION

This paper presents a new algorithm for detecting and protecting against DNS Amplification attack called PSVR-DNS. In the experimental part of the paper, the new PSVR-DNS algorithm is proposed and compared with the existing algorithms, the DNS Injector Detector and a Simple tool to detect. Comparison is conducted based on the time necessary for the attack detection, the attack detection error, the packet drop rate, the time required for attacker isolation, and the attacker isolation error. The time to detect a DNS Amplification attack of new PSVR-DNS algorithm of 25 seconds is, compared to the existing DNS Injector Detector algorithms time of 170 seconds, and a Simple tool to detect time of 235 seconds. In the test case with the 30 hosts (IoT nodes), the DNS Amplification attack detection error is analyzed. The smallest attack detection error (of approximately 39%) was provided by the new PSVR-DNS algorithm compared to existing algorithms, the DNS Injector Detector (53%), and a Simple tool to detect (55%). From the presented results, it can be concluded that the increase in the number of hosts (IoT nodes) simultaneously increases traffic and algorithms increase the error rate when detecting DNS Amplification attacks. When it comes to packet drop rate, the new PSVR-DNS algorithm had a minimal packet drop rate (of approximately 6%), compared to existing DNS Injector Detector algorithms (16%), and a Simple tool to detect (19%), because the new PSVR-DNS algorithm detects the attacker faster. When compared to the time required to isolate an attacker, the new PSVR-DNS algorithm takes at about 9 seconds to isolate an attacker. The existing DNS Injector Detector takes about 180 seconds, and a Simple tool to detect takes about 192 seconds. When the attacker isolation error is considered in the scenario with 30 hosts (IoT nodes), the new PSV-DNS algorithm provided the smallest error (of approximately 33%) compared to the existing algorithms, the DNS Injector Detector (82%), and a Simple tool to detect (84%). It can be concluded that the new PSVR-DNS algorithm provided better results when compared to existing algorithms because it was based on machine learning, SDN manager, and Bayesian probability that help in faster detection and isolation of DNS attacks. In the future work, the effects of different types of attacks on the proposed new PSVR-DNS algorithm will be considered.

6 REFERENCES

[1] Centar informacijske sigurnosti (2011). *Zaštita od prisluškivanja mrežnog prometa*. Zagreb. 1-23. https://www.cis.hr/files/dokumenti/CIS-DOC-2011-10-029.pdf

[2] Davies, K. (2008). Dns cache poisoning vulnerability explanation and remedies. *Internet Assigned Numbers Authority*, 1-43.

[3] Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., & Rajarajan, M. (2013). A survey of intrusion detection techniques in cloud. *Journal of networkand computer applications*, *36*(1), 42-57. https://doi.org/10.1016/j.jnca.2012.05.003

[4] Divya, C. & Francis Xavier Christopher, D. (2019). Security against arpspoofing attacks using bayesian support vector regression. *International Journal of Innovative Technology and Exploring Engineering*, *8*(7), 1-9.

[5] Chu, W., Keerthi, S., & Ong, C. J. (2004). Bayesian support vector regression using a unified loss functions. *IEEE transactions on neuralnetworks*, *15*(1), 29-44. https://doi.org/10.1109/TNN.2003.820830

[6] Jin, Y., Tomoishi, M., & Matsuura, S. (2019). A detection method against dns cache poisoning attacks using machine learning techniques: Workin progress. *2019 IEEE 18th International Symposium on Network Computing and Applications* (*NCA*), 1-3. https://doi.org/10.1109/NCA.2019.8935025

[7] Ma, T., Xu, C., Zhou, Z., Kuang, X., Zhong, L., & Grieco, L. A. (2020). Intelligent-driven adapting defense against the client-side dns cache poisoning in the cloud. *GLOBECOM 2020 IEEE GlobalCommunications Conference*, 1-6. https://doi.org/10.1109/GLOBECOM42002.2020.9322430

[8] Wu, H., Dang, X., Zhang, L., & Wang, L. (2015). Kalman filter based dns cache poisoning attack detection. *IEEE International Conferenceon Automation Science and Engineering* (*CASE*), 1594-1600.

[9] Hmood, H. S., Li, Z., Abdulwahid, H. K., & Zhang, Y. (2015). Adaptive caching approach to prevent dns cache poisoning attack. *The Computer Journal*, *58*(4), 973-985. https://doi.org/10.1093/comjnl/bxu023

[10] Wu, H., Dang, X., Wang, L., & He, L. (2016). Information fusion-based method for distributed domain name system cache poisoning attack detection andidentification. *IET Information Security*, *10*(1). https://doi.org/10.1049/iet-ifs.2014.0386

[11] Tzur-David, S., Lashchiver, K., Dolev, D., & Anker, T. (2012). Delay Fast Packets (DFP): Prevention of DNS Cache Poisoning. *SecureComm 2011: Security and Privacy in Communication Networks*, *96*(1), 308-318. https://doi.org/10.1007/978-3-642-31909-9_17

[12] Satam, P., Alipour, H., Al-Nashif, Y., & Harir, S. (2015). DNS-IDS: Securing DNS in the Cloud Era. *International Conference on Cloud and Autonomic Computing*. https://doi.org/10.1109/ICCAC.2015.46

[13] Evans, M. J. & Rosenthal, J. S. (2009). Probability and Statistics. *The Science of Uncertainty*, University of Toronto.

[14] Prabadevi, B. & Jeyanthi, N. (2018). TSCBA A Mitigation System for ARP Cache Poisoning Attacks. *Cybernetics and Information Technologies*, *18*(4). https://doi.org/10.2478/cait-2018-0049

[15] Ali Al-Mafrachi, B. H. (2017). Detection of DDoS Attacks against the SDN Controller using Statistical Approaches. *Wright State University*. https://etd.ohiolink.edu/apexprod/rws_etd/send_file/send?accession=wright1513738941473344&disposition=inline

[16] Etman, M. A. A. (2018). *DDoS Attack Detection System Using Semi-supervised Machine Learning in SDN*. Toronto Metropolitan University. Master Thesis. https://doi.org/10.32920/ryerson.14657868.v1

[17] Ahmad, F., Pervez, N., Junaid Arshad, M., & Ali, R. (2018). Detection and Mitigation of DDoS Attacks in Software Defined Networks. *International Journal of Multidisciplinary Sciences and Engineering* (*IJMSE*), *9*(5).

Contact information:

Denis PEJIĆ, MSc
(Corresponding author)
Faculty of Electrical Engineering, Computer Science and Information Technology,
Ul. Kneza Trpimira 2b, 31000, Osijek, Croatia
E-mail: denis.pejic@student.ferit.hr

Višnja KRIŽANOVIĆ, PhD
Faculty of Electrical Engineering, Computer Science and Information Technology,
Ul. Kneza Trpimira 2b, 31000, Osijek, Croatia
E-mail: visnja.krizanovic@ferit.hr

Drago ŽAGAR, PhD
Faculty of Electrical Engineering, Computer Science and Information Technology,
Ul. Kneza Trpimira 2b, 31000, Osijek, Croatia
E-mail: drago.zagar@ferit.hr