Protection of Network Security Selector Secrecy in Outsourced Network Testing

Sultan Alasmari
CS and IT College
Majmaah University
Kingdom of Saudi Arabia
su.alasmari@mu.edu.sa

Weichao Wang
Computing & Informatics
Univ. North Carolina Charlotte
Charlotte, NC
weichaowang@uncc.edu

Aidong Lu
Computing & Informatics
Univ. North Carolina Charlotte
Charlotte, NC
aidong.lu@uncc.edu

Yu Wang
Dept. of CS
Temple University
Philadelphia, PA
wangyu@temple.edu

***Abstract*—With the emergence and fast development of cloud computing and outsourced services, more and more companies start to use managed security service providers (MSSP) as their security service team. This approach can save the budget on maintaining its own security teams and depend on professional security persons to protect the company infrastructures and intellectual property.**

However, this approach also gives the MSSP opportunities to honor only a part of the security service level agreement. To prevent this from happening, researchers propose to use outsourced network testing to verify the execution of the security policies. During this procedure, the end customer has to design network testing traffic and provide it to the testers. Since the testing traffic is designed based on the security rules and selectors, external testers could derive the customer network security setup, and conduct subsequent attacks based on the learned knowledge. To protect the network security configuration secrecy in outsourced testing, in this paper we propose different methods to hide the accurate information. For Regex-based security selectors, we propose to introduce fake testing traffic to confuse the testers. For exact match and range based selectors, we propose to use NAT VM to hide the accurate information. We conduct simulation to show the protection effectiveness under different scenarios. We also discuss the advantages of our approaches and the potential challenges.

***Index Terms*—Outsourced network testing, Security selector secrecy, Prevent information leakage**

I. Introduction

With the fast development and wide deployment of cloud computing and light-weight container technology, organizations can focus more on their essential business and outsource most other operations. Within those outsourced operations, cyber security enforcement is a special case since it is not easy to verify whether or not a security service provider actually conducts the security scans or checks [1]. To verify the outsourced security operations, researchers have designed different methods to test their execution [2]–[5]. One of the proposed approaches involves recruitment of a large number of third party testers to send out carefully crafted network packets to verify the security rules enforced by the managed security service providers (MSSP) [6].

While such testing can effectively verify selected security rules, a severe concern was the disclosure of the network security policies and configurations. Specifically, in [6], the authors proposed to build a third-party platform, which would then recruit a large number of testers to send out network testing packets. Since the testing packets are carefully designed to verify the security rules, the tester can reversely derive the configuration. Through such reverse derivation (and maybe data ggregation among multiple testers), the testers may figure out the network security setup of the target environment, and design subsequent targeted attacks.

Below we will show an example. Assuming that a malware for a specific version of Windows has the payload signature of s_1. To test whether or not the outsourced network security service can detect such malware, the customer can create a network packet with the payload signature and request a third party tester to send the packet. Through this operation, the tester can learn the following facts: (1) the target IP address of the packet is probably running that specific version of Windows. (2) The customer is already aware of the malware and its signature. (3) The port number of the target machine is open for network traffic. While in real life the customer may manipulate the network testing traffic to mislead testers (we will study the impacts later in the paper), the tester can still learn some information about the network security setup, especially if multiple testers pull their information together.

This example illustrates the challenges that we face. When a customer outsources security testing to a large number of testers, it has to disclose certain amount of information to external players. This procedure is different from using a cloud security company or using a penetration testing company for the organization. For the former case, the cloud security company works with the customer to design and implement all network security mechanisms, so it already has full knowledge about the security setup. In the latter case, the penetration tester usually needs to sign an NDA (non-disclosure agreement) with the customer so that non of the testing results or detected vulnerabilities will be disclosed. Neither of the situations can fit into the approach in [6].

In this paper, we plan to investigate the questions described above, specifically the secrecy protection to the network security rules during the outsourced network testing operations. To resolve such challenges, we need to: (1) classify the network security selectors that we need to protect, and identify methods through which information leakage could happen during outsourced network testing; (2) based on the selector

classification results, design different methods to protect the secrecy of network security rules; and (3) through experiments, evaluate the trade-off between improved secrecy protection, test coverage, and increases in overhead.

Our research efforts will make the following contributions to the domain. First, as previous approaches focus on the overall architecture design and incentivization models of outsourced network testing, protection to network security configuration secrecy for customers has not been fully studied. This research will allow customers to weigh the benefit of outsourced services against potential threats to its information secrecy. Second, through classification of the security selectors and the design of methods for their protection, we investigate the outsourced network testing problem from a new aspect. Last but not least, the experiment results provide references for end users to consider trade-offs between outsourced services and scerecy concerns.

The remainder of the paper is organized as follows. In Section II, we will introduce related work, especially on the outsourced network security testing and privacy protection for such approaches. In Section III, we will introduce the basics of outsourced network testing, the application scenarios of our approaches, and the challenges we face. In Section IV, we will introduce the classification of the security selectors and the proposed mechanisms for their protection. We will discuss the pros and cons of each mechanism. In Section V, we will present the experiment results and the impacts of the proposed approaches on the cost and overhead of the network testing operations. Finally, in Section VI, we conclude the paper and discuss future extensions.

II. Related Work

With the fast development of entrepreneurship, more and more small to medium sized companies are created for business in niche markets. Such companies usually do not have a cybersecurity team with needed expertise. Therefore, it is quite popular for them to outsource the infrastructure and data security services [7]–[9]. To maintain a balance between the data and operation confidentiality and cost-effectiveness and safety of the companies, investigators have designed different mechanisms to identify the essential functions to outsource, mechanisms to verify the execution of the security functions, and protection to data privacy. Below we will discuss the existing work in these domains.

Outsourcing of Cooperation Security Functions

A modern cooperation needs to handle multiple aspects of cyber security, such as network security, access control, data confidentiality, and information privacy. Depending on the nature of a business, a company often needs to choose the security functions to outsource. For example, in [5], the authors study the security functions and show that the management of security outsourcing depends on the security efforts of both the managed security service provider (MSSP) and the customers, and their allocation of efforts can change dynamically during the contract period. The research also shows that a third party can serve as the cyberinsurance company to monitor the outsourced security function when the cost is low. In [10], the authors propose to outsource the data decryption function so that attribute based access control can be achieved more efficiently while keeping the same level of security in edge network in Internet of Vehicles. In [3], the managed security services (MSS) are modeled as bilateral liability-based contracts. The researchers design two novel contracts: threshold-based liability contract and variable liability contract, so that better results can be achieved when we can verify the data breaches.

Monitoring and Verifying Security Functions

The concept of Managed Security Service Providers (MSSP) is first used to achieve secure data sharing in large scale corporations [11]. MSSP often offers a platform solution for cyber threat monitoring and analytics, and conducts the operations such as firewalls, anti-virus services, and intrusion detection systems. Examples of such platforms include BT Cyber Security Platform [12], threat monitoring and detection software from BT and SAP, and the Enterprise Security Manager from McAfee. Since different platforms often label security incidents' priority differently, researchers have also studied relationship among cross-vendor incident indicators [13]. The EU's PALANTIR framework [14] tries to pull together the cyber-security intelligence from both large corporations and small, medium business to enable more efficient and effective attack detection and mitigation. The platform supports different ways to host the security capabilities, and integrates the threat intelligence, capability orchestration, and attestation and recovery capabilities to enforce security and resilience.

Another series of research focuses on outsourcing of security functions and verification of security SLA (service level agreement). In [2], the researchers propose to verify the outsourced data encryption operations by third parties. The authors define the expected properties of such verification operation, and investigate the relationship between the customer, tester, and security service provider. In [6], the authors push the application domain to the general network security operations and propose to build a platform to recruit and monitor the testing operations. In [15], the authors integrate fully homomorphic encryption with polynomial factorization algorithm to support public verification on the computation result while protecting data security.

Privacy Preservation in Security Function Outsourcing

Outsourcing of data or infrastructure security functions usually needs to disclose sensitive data or company information to third parties. Therefore, research efforts have been conducted to assess such disclosure and protect data ownership. In [16], the authors build a game theory model to compare in-house security enforcement with outsourced security operations. They suggested that partial outsourcing can help corporations achieve a balance between cost efficiency and data security. Their efforts also pave the way of integrating the information leakage assessment into the game theory model.

In [17], the author shows that when event logs with personally identifiable information (PII) are released to external security operation center (SOC), user privacy will be violated. The paper extended beyond previous efforts that focused only on IP address and investigated different methods of pseudonymization to hide the sensitive information. While the non-deterministic methods provide higher level of privacy protection to end users, the utility of the data is also affected since the association among log records is weakened.

In [18], the author used differential privacy guarantees as guard for user data, and designed secure data outsourcing, integration, and query mechanisms. The methods allow publicly verifiable results while preserving the data privacy. The author used Bitcoin transactions as the application scenario. And the end users can verify the digital currency transactions while keeping the transaction holding and total holding of the participating parties.

In [19], the authors proposed to use homomorphic encryption to support the outsourced data so that query and processing can be conducted while the plain text of data is kept from the managed security services (MSS). The challenges include the computation overhead, especially when the mechanism is deployed in a large scale environment, and the types of supported operations using homomorphic algorithms.

Based on the discussion, we can see that some of the methods, especially the assessment of information leakage, can benefit our research on the protection to the security configurations secrecy at the end customers. Below we will describe our application scenarios and the design of several methods for the research goals.

III. Application Scenarios and Challenges to Information Secrecy

A. Application Scenario

Figure 1 shows the basic working scenario of the outsourced network testing. To verify that the MSSP actually enforces the negotiated security services, the end customer U allows the external third parties to send network testing traffic to assess the network security policies. To accurately assess the policies, the customer U needs to design the testing packets diligently. For example, assuming that MSSP is supposed to check the signature s_1 of a known malware in the packet payload. The customer needs to embed s_1 into the testing packet and based on whether or not it can receive the packet from the tester, it can determine whether or not the content based security rule is enforced.

While such operations allow the verification of security enforcement by MSSP, they also disclose sensitive information about U. For example, the case above tells the tester that "U is aware of the attack and avoid using it for compromise attempt". To generalize the scenario, if the packets are used to test the open ports on existing IP addresses on the customer network, it actually becomes a weaker version of netscan for the testers.

This analysis shows that certain methods must be designed and adopted to help U maintain a balance between the effectiveness of outsourced network testing and the secrecy of its network security configurations. While there have been different ways to classify the network security policies, in this paper we use the classification described in [20]. Specifically, we consider two types of security policies and their leakage prevention. Although these two types do not cover all possible classification methods, we choose them since they demand different ways of protection.

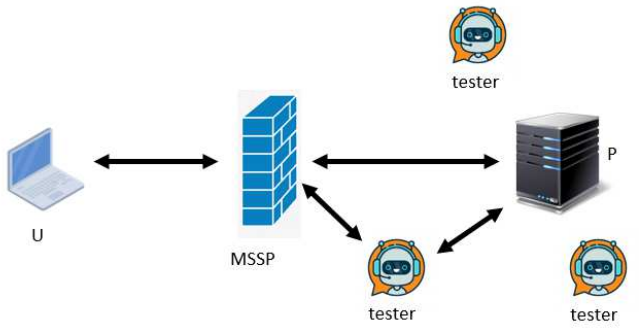


Fig. 1. Application scenarios of the proposed approach. Courtesy of authors of [6].

The first type of security selectors covers the 'exact match selector' and 'range-based selector' described in [20]. These are the frequently seen security policies that use IP addresses or port numbers as filters for equality check or range check. The IP address prefix also falls into this type.

The second type covers the 'regex-based selector' and 'customer check selector' in [20]. An example of the former selector is string match such as URL links. An example of the latter selector includes the specific patterns used by malware analysis and intrusion detection systems (IDS) tools, as the case described at the beginning of this section.

B. Threat Model and Leakage Prevention Challenges

We assume that the outsourced network testers are recruited by the testing platform P. The primary criteria for tester recruitment is the trustworthiness on their faithful execution of the testing tasks. Specifically, the testers will send out the testing traffic as the customer requests. Their other usage of the testing traffic is a less concern. Therefore, in this paper we adopt the curious but not malicious model for the testers. Specifically, we assume that the testers will not actively generate network probing packets to the customer to explore its network configuration and potential vulnerabilities. On the contrary, the testers will collect useful information from the testing traffic and try to derive network security configurations.

As described in [2], the network testing traffic must be sent by different testers to prevent the MSSP from identifying the packets and treating them differently. Some researchers may consider this as a natural protection to the network security configuration since each tester can get only a small share of the information. However, since the testers could collaborate offline to put their information together, the testing platform cannot tell which nodes are actually sharing information among them, and how much information a group of testers can

actually learn. Therefore, some mechanisms must be designed to protect the information secrecy.

IV. Preventing Information Leakage during Outsourced Network Testing

Through previous analysis, we can see that special mechanisms must be designed to prevent information leakage through the outsourced network testing. Our initial study shows that different types of security policies need different schemes for protection. Below we will describe the mechanisms and analyze the protection effectiveness.

Since we focus on the security selectors discussed in [20], in Figure 2 we show some examples in each type. The operations that the MSSP adopts will be determined by the rule set. Here the customer could directly provide the selector rules to MSSP, or it can provide only the functional and security requirements from the high level, and the MSSP can then use the policy language [21], [22] to accomplish policy construction and orchestration. To verify whether or not the security policies are accurately executed, the customer can work with the testers to design and deliver the testing traffic.

Type 1 selector		Type 2 selector	
Exact match rules:		Regex based rules:	
(1) Dest IP = 135.5.4.35	Op1	(1) URL = www.utoday.org	Op4
(2) Src port = 23	Op2	- - -	
- - -		Custom check rules:	
Range based rules:		(1) Data hash = 0xff45 0067	Op3
(1) Src port in (1000, 1050)	Op2	- - -	
(2) Src IP in 156.5.x.x	Op3		

Fig. 2. Example rule table at the MSSP.

A. Protecting the Regex-based Selectors

In this subsection, we will focus on the protection of regex-based selectors. Here the security rules are usually string based or binary value based. The MSSP will examine the contents of the packets and make corresponding operations based on the rules.

Since the security rules are dynamic, the customer U can continuously add new rules to the database. Therefore, it is essential to keep the secrecy of the database contents from external parties so that we can deter or reduce the impacts of the attacks. However, to verify whether or not the MSSP is actually enforcing the security rules, the testing traffic needs to be designed based on the active rules.

To solve this dilemma, we propose to introduce fake security selectors into the database and construct the testing traffic based on these selectors as well. Specifically, through introducing the fake links of malicious webpages or binary content signatures, we can test the enforcement of security rules by MSSP while preventing the testers from learning our real security policies.

As a specific example, assuming that the customer U constructs a fake signature sig_m of a non-existent malware and introduces it into the selector database. It will then construct a testing packet based on the selector and provide it to the tester. From the tester's point of view, since it does not recognize the signature, it will treat it as some new knowledge that it learns about the customer U. It will not impact the execution of the testing activity since it just needs to send the packet and conduct subsequent interactions if needed. From the MSSP's point of view, dynamic updates to the security selector database is also a very normal operation. If it is honest and faithfully executing the operations, it just needs to take suggested actions based on the rules.

A.1 Challenges and Methods

While it seems to be straightforward to add or remove fake security selectors to protect the secrecy of security rules, we still face some challenges. First, what is the impact of fake selectors on the effectiveness, efficiency, and cost of outsourced network testing? Second, how will the fake selector impact the flow of normal traffic? Specifically, will the fake signatures lead to silent discard of benign data? Last but not least, can the MSSP or the tester distinguish a real selector from a fake selector? If so, how will they handle the packet differently? Below we will discuss these questions.

First, let us investigate how the size of the fake selectors impacts the efficiency and cost of the outsourced network security services. Based on a report released by RSI Security [23], MSSP usually charges the price of $75 to $250 per user per month based on the size of the company and the complexity of the business functions. Some popular third party security services, such as AWS Network Pricing, charge the customers based on the number of firewall endpoints and the amount of traffic [24]. As each Azure Network Security Group (NSG) can have a maximum of 1000 rules, introducing a certain number of fake selectors will not impact the price to a large extent.

Research in [25] shows that as the number of rules in a firewall increases, the overall performance will decrease. With the adoption of certain methods such as segmentation and rule indexing, we can make the impact acceptable. For example, when the number of rules increases from 1 to 1000, the throughput will decrease by about 6% [25]. However, when the number of rules increases to 10000, the throughput will decrease by about 75%. From this data, we can see that when we do not add too many fake selectors, the performance impact is acceptable.

The second challenge we face is the impact of the fake selectors on benign traffic. For example, if the customer introduces a fake binary signature sig_m into the selector database, there is a chance that some benign data will match to the signature and get dropped. As the number of fake selectors increases, the probability that a benign data has signature collision also increases. To reduce the probability of signature collision with benign traffic, we propose the following approaches. (a) use long hash results to label binary signatures of the selectors. Using a long hash results (e.g. 128 bits instead of 64 bits) will drastically reduce the probability of random collision. (b) Since addition and removal of selectors can be dynamically conducted, the customer can choose to add

the fake signatures into the database and provide the testing packets to t. After the testing operations, it can then remove the fake signatures. Since this procedure is transparent to the testers, they cannot learn about the changes of the database.

The third challenge we face is the fake URL links since they are different from the fake binary signatures. When a curious tester receives a binary signature that it does not recognize, it will assume that this is a new malware. However, this cannot be applied to fake URL for selectors. If the customer chooses to use a real URL as a selector, any traffic targeting at that website will be filtered out. This will impact the normal network operations. If the customer uses a fake URL, the tester can easily identify it by surfing the Internet.

To address this challenge, we propose to use the policy-level capability proposed in [20]. Specifically, we propose to design an abstract decision criteria for the action to apply only when the testing packet satisfies two or more conditions simultaneously. For example, while we can add a real URL as the selector, a quadruple consisting of (the source IP and port, and the destination IP and port) can be used as the second criteria to avoid packet discard by accident. In this way, the tester could not link the two conditions together to identify the fake selector.

A.2 Quantitative Results

As the discussion above shows, we can add a group of fake selectors into the security database and use them as the test cases to preserve the secrecy of the network configurations. In this part, we will conduct some analysis to show that under different cases, what a tester can learn from the test traffic generated by the customer.

We assume that customer U has r real security selectors. To protect the secrecy of network security configuration, U also introduces f fake selectors. To verify the execution of the security rules by MSSP, U will choose t_1 rules from r and t_2 rules from f to construct the testing traffic. When the tester receives the constructed packets from U, it will try to figure out which rules are real and which rules are fake. To present the analysis results in a more conservative way, we assume that the tester knows the ratio between t_1 and t_2. In real life, of course, the customer U will not share the ratio with any third party.

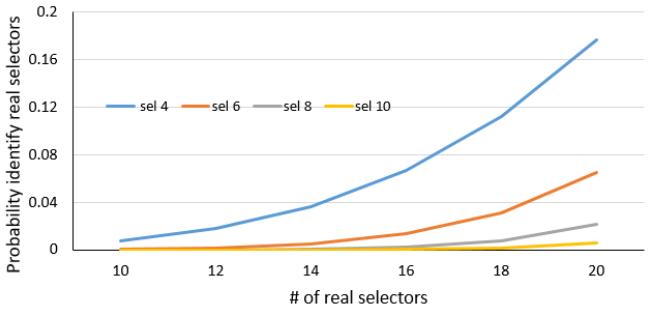
Here we will present two scenarios. In the first scenario, the tester will select up to t_1 packets provided by U and hope that all of them are from r. In the second scenario, the tester will also choose multiple packets from U and hope that a portion of them are drawn from r. Note that the two cases represent the optimal result and a more practical result for the tester, respectively.

Case 1: In this case, the tester will choose p ($p = 1$ *to* t_1) packets from the $(t_1 + t_2)$ packets provided by U and hope that all of them are real. The probability is $C\binom{p}{t_1}/C\binom{p}{t_1+t_2}$.

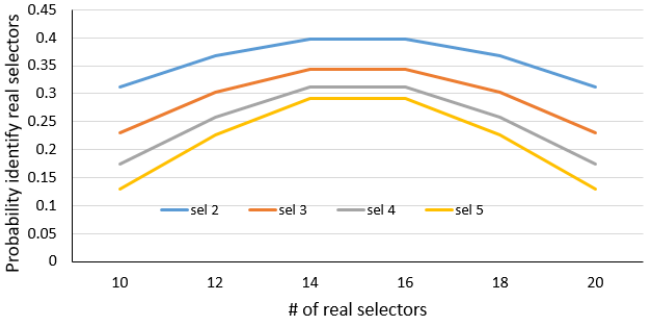
Case 2: In this case, the tester will choose p ($p = 1$ *to* $t_1 + t_2$) packets from the $(t_1 + t_2)$ packets provided by U and hope that x_1 selectors are real while x_2 selectors are fake. Here $(x_1 + x_2) = p$. At this time, the tester will not know which ones it selects are real. Therefore, the probability is $C\binom{x_1}{t_1} * C\binom{x_2}{t_2} / C\binom{x_1+x_2}{t_1+t_2}$.

Figure 3 shows the simulation results. As a special case, we choose $t_1 + t_2 = 30$ (thus the tester will receive 30 testing packets from U and it tries to identify which are real). Here the number of packets constructed based on real selectors t_1 ranges from 10 to 20, and the fake selectors are (30 - t_1). We assume that the tester chooses p's value between 4 and 10. The following two figures show the results.

In Figure 3.(a), on the X-axis, we show the number of real selectors that the customer sends out. Therefore, the number of fake selectors is (30 - X-axis). The four lines in the figure show the number of selectors that the tester picks from the packets that the customer provides. For example, the blue line shows the situation in which the tester selects 4 packets provided by the customer. On the Y-axis, we show the probability that the packets selected by the tester are all real selectors.



(a) Probability that all identified selectors are real.



(b) Probability that half of identified selectors are real. But do not know which half.

Fig. 3. Simulation results of the selector identification accuracy at the tester node.

From the figure, we can see that as the number of real selectors increases in the provided packets, the tester has higher probability to select them. However, even when the customer puts 20 real selectors into the 30 packets, the probability that all four of the selected packets are real is still smaller than 20%. As the number of selected packets increases, the probability decreases fast. For example, when t_1 = 20 and p = 10, the probability is smaller than 1%.

In Figure 3.(b), the X-axis has the same set of values. However, when the tester selects packets from the set, it expects half of the selectors are real while the other half are

fake. For example, the blue line shows 'select 2' which means 2 of the selected packets are real and 2 are fake (in total the tester still selects 4 packets). On the Y-axis, we have the probability.

From the figure, we can first see that the lines are symmetry. This is caused by our experiment setup in which the selected packets will contain half real and half fake. The second observation we have is that the probability is much higher in Figure 3.(b) than that in Figure 3.(a). This is also reasonable since now the tester will choose from two pools (one fake and one real) and there are more combinations of the selectors. However, we have to emphasize that even though the tester has a much better chance to have half of the selectors to be real, it cannot differentiate real selectors from the fake ones. Therefore, it still faces difficulties in deriving knowledge about the customer network and conducting attacks.

B. Protecting the Exact Match and Range Selectors

Protection of the exact match and range based selectors are more challenging. First, while we can still introduce fake selectors into the database, it will be much easier for the MSSP to figure out that this selector is fake. For example, the customer may configure a fake rule stating that "All incoming traffic to IP address a.b.c.d will be silently droped". However, it will not be hard for the MSSP to figure out that there is no machine in the customer's network that owns the specific IP address. Therefore, the sender of a packet satisfying the rule can be labeled as a 'tester'. As a counter example, in the Regex based selectors, it is much more difficult for the MSSP to figure out whether or not a binary signature of a virus actually exists. For the similar reason, we cannot just use the abstract decision criteria that we discussed in above sections since the combination of multiple criteria can prevent a rule from being triggered by external traffic, but it cannot prevent the MSSP from figuring out the fake address.

The second challenge we face is on the information leakage through specific port numbers. We know that certain network services are associated with well accepted port numbers. Therefore, when testing traffic is provided to a tester for a specific (IP address, port) combination, the tester can easily figure out that this is a server for the application.

The third challenge we face is the information leakage through the test of range based selectors. To verify that the MSSP actually implements the range based rules instead of examining individual values, we need to construct multiple testing packets for a rule. This scheme, however, could leak information on the value range.

To respond to these challenges, we propose to dynamically deploy a NAT (Network Address Translation) VM in front of the MSSP gateway to accomplish address hiding and replacement. While the NAT technology was originally designed for home networks and small scale enterprise networks since they often do not have an enough number of globally recognized IP addresses, it is still widely adopted in the modern cloud environment. For example, in the EdgeVPN.io environment [26], the authors proposed a deployment of virtual networks spanning across distributed edge and cloud resources. Since the edge nodes could be deployed in a large distributed area, they will often be assigned private addresses. Therefore, the NAT middle-boxes are used to connect them through a virtual network. A similar technique was also presented in [27].

As shown in Figure 4, the NAT VM is deployed in front of the MSSP server. The customer will configure the rules of the NAT based on the generated packets. Specifically, since the customer will know the IP address of the tester, it can set up the triggering condition of a certain rule by identifying the (src IP, src port, dest IP, dest port) tuple. In this way, only the specific testing packets will trigger the rules. Note that the rules need to be configured in dual directions for multi-round testing. When such a rule is triggered, the NAT box will replace corresponding fields in the packets so that selectors in MSSP can be activated. This procedure, however, is transparent to the tester.

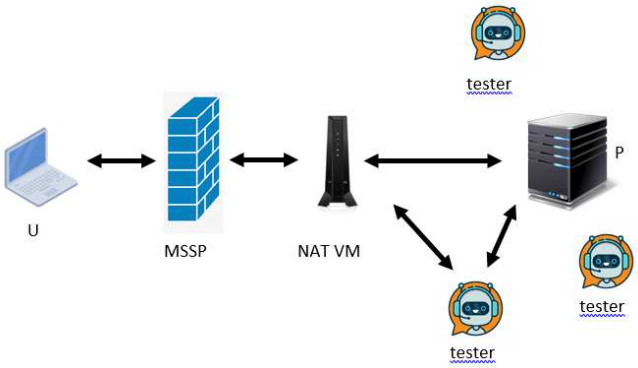


Fig. 4. Protecting the exact match and range based selectors through NAT VM.

Such configuration will not impact the normal operations of the customer network. We will provide an example. Assume that the customer provides a service at the IP address a.b.c.d to a certain group. The tester, however, does not belong to the group. Therefore, during the network testing procedure, the customer does not want to disclose the real IP to the tester. To achieve the goal, it will configure the NAT for the traffic between the tester and the customer. From the tester's view, it is really communicating with the fake IP address. From the MSSP's point of view, it only sees a packet for the normal service address.

B.1 Robustness against Traffic Analysis by Testers

In this part, we are going to discuss the robustness and hiding capability of the NAT VM to protect the network security configuration of the customer network. In past years, machine learning algorithms have been adopted to analyze the traffic patterns and other properties to identify the IP address clusters. For example, in [28], the authors proposed to use machine learning algorithms to identify and count total number of hosts behind NAT. Their primary goal is to figure out the number of nodes behind one or a small group of IP addresses that are conducting the DoS attacks. The basic idea is to analyze patterns in network traffic through flow level statistics.

As another example, in IPvest [29], the authors also tried to identify the number of nodes behind a NAT and cluster their traffic based on the patterns. The system will first use the features such as time and duration, traffic direction and amount, and http-cookies to label the flows. They then build a DGA (dynamic grid algorithm) to discover the number of nodes. Specifically, the time and OS related features are used jointly to figure out the number of devices. They then use the Gaussian mixture modeling (GMM) and Ward's agglomerative algorithm (Ward) to accomplish the traffic flow clustering.

From the discussion, we can see that the ML algorithms need to have access to a large volume of data from the node groups for classification and clustering. In our application scenarios, through controlling the number of testing traffic to individual testers, we can reduce the machine learning output accuracy on the node identification and traffic clustering. At the same time, since all the testing traffic is carefully crafted by the customer, it can use a different pattern from that of its normal traffic to better hide the connections.

V. Evaluation of the Proposed Approaches

In the previous sections, we have introduced the mechanisms to protect the secrecy of the network security configuration at the customer from the testers. Since the MSSP is helping the customer to secure the network, we do not need to protect the information from it. However, we still do not want the MSSP to differentiate testing packets for real security selectors from those for the fake ones since the MSSP can treat the packets differently to pass the verification.

In this part we will evaluate the proposed approaches. There are multiple aspects from which we can investigate the impacts of the approaches on the network performance, cost of testing, information collection and derivation at testers, and the effectiveness of the approaches. Below we will first present the evaluation setups, and then the details of the results.

A. Experiment Settings

During the generation of the network testing traffic, a part of the packets will be generated based on the real selectors, while the remaining parts of the testing packets are built upon fake selectors. Here we assume that when we construct the packets for real selectors, the selectors are picked based on a uniform distribution. In this way, we can prevent the MSSP from predicting which selectors have higher probability to be verified.

For the test of fake selectors, we will consider two scenarios. In the first scenario, the customer will dynamically generate the group of fake selectors and insert them into the database of the MSSP. It will then construct testing traffic based on these dynamically generated rules. Since these selectors are generated to mislead the testers, they will be removed from the database after usage. At the same time, since the fake selectors are dynamically generated when they are needed, the network security configuration represented by the rules demonstrate an ad hoc format. In other words, the fake selectors and their testing packets may contain conflicting information. For example, when a tester cross compares two sets of testing packets, it may find that the customer is adopting two conflicting decisions on the same URL, which could lead to the identification of fake selectors.

For the second scenario, the customer will adopt the deep fake policies. Specifically, the customer will construct a group of selectors based on a pre-generated fake network topology and its security configuration. From an external party's point of view, it seems that this component of network actually exists and the security rules are aligned to the network topology and functionality. It will become much more difficult for the testers to identify the fake selectors even when they pool the testing traffic at multiple testers together. The cost of the deep fake, however, includes the generation and maintenance of the network topology and its configuration, and the longer stays of the rules in the database. The testing traffic for the exact match and range based selectors upon the fake topology needs to pass through NAT VM to prevent the MSSP from identifying the testers.

B. Storage and Processing Overhead

The extra delay of the proposed approaches consists of two parts. In the first part, the NAT VM needs to compare the input and output packets with the rules to determine whether or not a change to the packet header is needed. The delay of this effort heavily depends on the layer of hardware and software in which such search and replacement is implemented. For example, the efforts in [30] show that if the operations are conducted in cache, it needs tens of microseconds to accomplish the tasks. Since our NAT VM usually does not need a large size to store the rules, we expect the similar length of delay.

The second source of extra delay is the increased database size at the MSSP. There have been extensive research on the performance of the firewalls and its relationship to the rule set size. For example, in [31], experiments show that with proper decomposition and composition, even when the rule set size increases for about 40 times, the delay will increase to two times. Therefore, we do not worry too much about the increased processing delay.

The other overhead we need to consider is the increased storage overhead since we introduce fake rules into the MSSP database. Such increase is linear to the number of fake rules that the customer generates. One thing we do not have space to explore extensively in this paper, is the change of the rule database internal structure caused by the insertion and deletion of fake rules. The previous research [31] shows that such reorganization could take multiple seconds before the structure becomes stable again. Future research can be conducted to evaluate the overhead in this aspect.

C. Aggregated Derivation of the Real Selectors

In this part, we will discuss the relationship between the derivation of the real selectors and the amount of testing traffic that is generated and distributed by the customer. First, we would like to emphasize that if the testers operate

independently, the customer can always deliver the testing packets for the same set of rules to the same node. Therefore, a tester can only learn from that group of packets. However, if multiple testers can put their knowledge together, they can learn from a much larger set, and identify the real selectors from different features such as the frequency that they are verified. In this way, they can use these selectors as targets of future attacks.

To illustrate the procedure, we conduct some simulation and investigate the aggregated derivation by the testers. Here we use the test frequency of different rules as an indicator. In the first simulation, we mimic the first scenario described in Section V.A. We assume that there are 100 real selectors and 800 fake selectors (in which the fake ones are about one degree of magnitude larger than the real ones). We also assume that the customer will conduct 100 times verification. For each verification it will choose 10 real selectors, and 5 or 20 fake selectors. Although in real situations we will choose as many testers as possible to increase the difficulty of information aggregation by them, here we assume that some parties will get access to all of the testing traffic. Below we illustrate the number of times that each rule is tested.

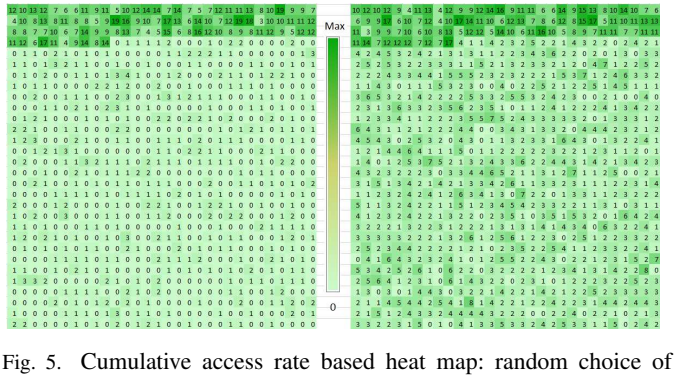


Fig. 5. Cumulative access rate based heat map: random choice of fake selectors.

Figure 5 shows the simulation results as a heap map. Note that we have 900 selectors altogether (30 x 30), and the first 100 are real ones (first 3.3 rows). In the left side, twice as many real selectors as the fake ones are chosen by the customer in each round of verification. The heat map shows the number of times that each rule is selected: the darker is the green color, the more times it is chosen. On the right side of the figure, the meaning of the color is the same, but in each round twice as many fake selectors as the real ones are chosen.

Based on the illustration, we can see that since the fake selectors have a much larger size and the chosen procedure is random, the real selectors have a much higher probability to be tested. If the testers aggregate their information, they can soon differentiate the fake ones from the real ones.

In the second simulation scenario, we make some adjustment. First, instead of generating fake selectors randomly, the customer will construct a set of fake selectors based on a pre-generated fake network topology, as we described in Section V.A. Therefore, the fake rules are also stable and will not contain conflicting information. Second, because of the complexity to construct and maintain the fake network topology, we will have only 300 fake selectors. Therefore, we will have a 20 x 20 matrix and the top 25% represent the real rules.

The simulation results are shown in Figure 6. Note that the first five rows contain the real selectors. Since the total number of selectors is much smaller in this scenario, the grids are larger. On the lest side, twice as many real selectors as the fake ones are chosen by the customer in each round of verification. On the right side, three times as many fake selectors as the real ones are chosen in each round of verification. Since the number of fake selectors is three times of the real ones, in the right side figure we cannot tell any difference only based on access frequency. Advanced mechanisms must be adopted by the testers to identify the real selectors.

When we compare the results in Figures 5 and 6, we can see that if the customer wants to prevent a tester from learning its network security configuration from the testing traffic only through the test frequency, it must control the sample rate of the selectors. At the same time, deep fake through the generation of a stable yet deceptive network topology and corresponding selectors will achieve muxh better protection to information secrecy.

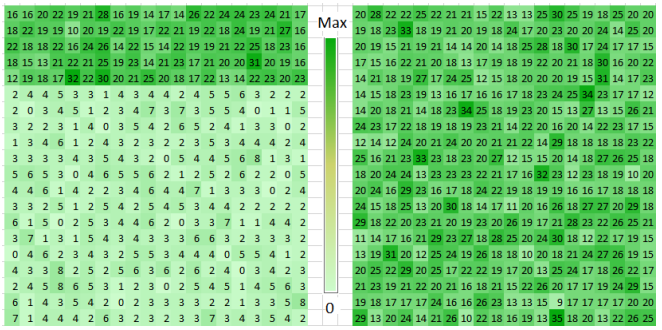


Fig. 6. Cumulative access rate based heat map: pre-generated fake network infrastructure.

D. Cost of the Protection

In [6], the authors have analyzed a model for the customer to achieve a balance between the coverage of verification of the selectors and the total cost. When we start to introduce fake selectors and testing traffic into the approach, the total cost will also be impacted. The impacts can be categorized as follows. First, although in [6] the authors adopt a linear cost model with the number of testing packets, in real life the testers could adopt a non-linear price, especially when it is getting close to the testing capacity. It will introduce more complicated constraints into the solution space and incur higher computation overhead.

The second impact is on the coverage of the real selectors that are tested. To control the overall test cost, the customer needs to control the number of testing packets. But the introduction of fake selectors will reduce the portion of real selectors that are tested. In this paper, we use only the frequency of access as the evaluator to differentiate the real

selectors from the fake ones. Since the testers could adopt more complicated ML algorithms to achieve the goal, the customer must adapt the selection procedure to protect the information secrecy.

VI. Future Extensions and Conclusion

A. Future Extension

In this paper, we introduce the problem of network security policy leakage caused by the outsourced testing. We use fake selectors and NAT VM to hide the real information. For the next steps, we plan to investigate from the following aspects.

Moving Target for Privacy Protection

Part of the reason that information leakage through the testing traffic is caused by the static network security configuration of the customer. If moving target defense mechanisms are adopted, the internal topology of the customer network will continue to change. Therefore, the information that the testers learned from the testing traffic will expire after each round of changes. The challenges, however, include the determination of the change frequency, maintenance of existing connections, and re-configuration of network devices and MSSP policies. The frequency of topology changes will also impact the frequency of outsourced testing.

Unified Cost-Coverage-Privacy Model

For the outsourced network testing, we have investigated the overall architecture, cost function, and incentivization of the testers through separate efforts. Our next step is to build a unified cost-coverage-privacy model for the outsourced testing efforts. The three factors will impact each other and it may create different choice criteria based on the needs of the customers. The model will also motivate the MSSP to better serve the security service level agreement (SSLA).

B. Conclusion

In this paper, we study the protection of the network security configuaration secrecy under the assumtion that the customer depends on outsourced network testing to verify the execution of the security policies. We first show the importance of such protection since the disclosed information could be used as network probing results and enable subsequent attacks. Based on the types of security selectors, we propose different protection approaches. For the Regex based selectors, we propose to use fake selectors to protect their secrecy. For the exact match and range based selectors, because of their unique features, we propose to use NAT VM to hide the accurate information from the testers. We conduct some simulation to investigate the advantages of the approaches. We also discuss the challenges we face.

Security service level agreement, as a special type of cloud services, is usually difficult to verify its execution. However, with the continuous development of vertical segmentation in technology, new corporations will depend more on the outsourced security service providers. Therefore, the verification of the implementation of the security policies will provide confidence in end customers on the safety and reliability of their business. The research direction deserves more efforts from a larger research population.

References

[1] S. Alasmari, W. Wang, and Y. Wang, "Proof of network security services: Enforcement of security sla through outsourced network testing," in *International Conference on Communication and Network Security (ICCNS)*, 2020.

[2] S. Alasmari, W. Wang, T. Qin, and Y. Wang, "Proof of outsourced encryption: Cross verification of security service level agreement," *Springer CCF Transactions on Networking*, pp. 229–244, November 2020.

[3] K.-L. Hui, P. F. Ke, Y. Yao, and W. T. Yue, "Bilateral liability-based contracts in information security outsourcing," *Information Systems Research*, vol. 30:2, pp. 411–429, 2019.

[4] A. Razaque, M. B. H. Frej, B. Alotaibi, and M. Alotaibi, "Privacy preservation models for third-party auditor over cloud computing: A survey," *Electronics*, vol. 10, no. 21, 2021.

[5] Y. Wu, G. K. Tayi, G. Feng, and R. Y. K. Fung, "Managing information security outsourcing in a dynamic cooperation environment," *Journal of the Association for Information Systems*, vol. 22, no. 3, 2021.

[6] S. Alasmari, W. Wang, and Y. Wang, "Incentivisation of outsourced network testing: View from platform perspective," in *the International Conference on Information Systems Security and Privacy (ICISSP)*, 2022.

[7] L. Coppolino, S. D'Antonio, G. Mazzeo, L. Romano, and L. Sgaglione, "Prisiem: Enabling privacy-preserving managed security services," *Journal of Network and Computer Applications*, vol. 203, p. 103397, 2022.

[8] H. K. Skrodelis, J. Strebko, and A. Romanovs, "The information system security governance tasks in small and medium enterprises," in *International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*, 2020, pp. 1–4.

[9] M. Zhao, G. Wei, C. Wei, and Y. Guo, "Cpt-todim method for bipolar fuzzy multi-attribute group decision making and its application to network security service provider selection," *Int J. Intell Syst*, no. 36, pp. 1943–1969, 2021.

[10] C. Feng, K. Yu, M. Aloqaily, M. Alazab, Z. Lv, and S. Mumtaz, "Attribute-based encryption with parallel outsourced decryption for edge intelligent iov," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13 784–13 795, 2020.

[11] X. Wang, I. Herwono, F. Cerbo, P. Kearney, and M. Shackleton, "Enabling cyber security data sharing for large-scale enterprises using managed security services," in *IEEE Conference on Communications and Network Security (CNS)*, 2018, pp. 1–7.

[12] BT, "Cyber security platform," https://www.globalservices.bt.com/en/solutions/solution/threat-detection-and-management-solutions, 2023.

[13] T. Shibahara, H. Kodera, D. Chiba, M. Akiyama, K. Hato, O. Söderström, D. Dalek, and M. Murata, "Cross-vendor knowledge transfer for managed security services with triplet network," in *Proceedings of the ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 59—-69.

[14] E. Mantas and et. al., "Practical autonomous cyberhealth for resilient micro, small and medium-sized enterprises," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, 2021, pp. 500–505.

[15] X. Yu, Z. Yan, and R. Zhang, "Verifiable outsourced computation over encrypted data," *Information Sciences*, vol. 479, pp. 372–385, 2019.

[16] N. Feng, Y. Chen, H. Feng, D. Li, and M. Li, "To outsource or not: The impact of information leakage risk on information security strategy," *Information Management*, vol. 57, no. 5, p. 103215, 2020.

[17] A. Rasic, "Anonymization of event logs for network security monitoring," Master's thesis, Concordia University, 2020.

[18] G. Dagher, "Secure protocols for privacy-preserving data outsourcing, integration, and auditing," Ph.D. dissertation, Concordia University, 2016.

[19] L. Sgaglione, L. Coppolino, S. D'Antonio, G. Mazzeo, L. Romano, D. Cotroneo, and A. Scognamiglio, "Privacy preserving intrusion detection via homomorphic encryption," in *IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 2019, pp. 321–326.

[20] C. Basile, F. Valenza, A. Lioy, D. R. Lopez, and A. Pastor Perales, "Adding support for automatic enforcement of security policies in nfv networks," *IEEE/ACM Transactions on Networking*, vol. 27, no. 2, pp. 707–720, 2019.
[21] Q. Kang, L. Xue, A. Morrison, Y. Tang, A. Chen, and X. Luo, "Programmable in-network security for context-aware byod policies," in *USENIX Security*, 2020.
[22] A. Molina Zarca, M. Bagaa, J. Bernal Bernabe, T. Taleb, and A. F. Skarmeta, "Semantic-aware security orchestration in sdn/nfv-enabled iot systems," *Sensors*, vol. 20, no. 13, 2020.
[23] RSI Security, "How much does it cost to outsource it security services?" https://blog.rsisecurity.com/how-much-does-it-cost-to-outsource-it-security-services/, 2021.
[24] AWS, "Aws network firewall pricing," https://aws.amazon.com/network-firewall/pricing/, 2023.
[25] M. Christiansen and E. Fleury, "An mtidd based firewall: Using decision diagrams for packet filtering," *Telecommunication Systems*, vol. 27, no. 2-4, pp. 297–319, 2004.
[26] R. Figueiredo and K. Subratie, "Demo: Edgevpn.io: Open-source virtual private network for seamless edge computing with kubernetes," in *IEEE/ACM Symposium on Edge Computing (SEC)*, 2020, pp. 190–192.
[27] M. Qutait, "Using software defined network and virtualization to implement nat and backup functionality and resulting business benefits of implementation in isps," http://dx.doi.org/10.2139/ssrn.3713860, 2020.
[28] S. Shukla and H. Gupta, "Identification and counting of hosts behind nat using machine learning," *SN COMPUT. SCI.*, vol. 3, no. 126, 2022.
[29] R. Mateless, H. Zlatokrilov, L. Orevi, M. Segal, and R. Moskovitch, "Ipvest: Clustering the ip traffic of network entities hidden behind a single ip address using machine learning," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3647–3661, 2021.
[30] G. Li, M. Zhang, C. Liu, X. Kong, A. Chen, G. Gu, and H. Duan, "Nethcf: Enabling line-rate and adaptive spoofed ip traffic filtering," in *IEEE International Conference on Network Protocols (ICNP)*, 2019, pp. 1–12.
[31] F. Nife and Z. Kotulski, "Application-aware firewall mechanism for software defined networks," *J Netw Syst Manage*, vol. 28, pp. 605—626, 2020.