Building a Cognizant Honeypot for Detecting Active Fingerprinting Attacks Using Dynamic Fuzzy Rule Interpolation

Naik, Nitin; Shang, Changjing; Jenkins, Paul; Shen, Qiang

Published in:
Expert Systems

DOI:
10.1111/exsy.12557

Publication date:
2021

Citation for published version (APA):
Naik, N., Shang, C., Jenkins, P., & Shen, Q. (2021). Building a Cognizant Honeypot for Detecting Active Fingerprinting Attacks Using Dynamic Fuzzy Rule Interpolation. *Expert Systems* , *38*(5), Article e12557. https://doi.org/10.1111/exsy.12557

Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ARTICLE TYPE

Building a Cognizant Honeypot for Detecting Active Fingerprinting Attacks Using Dynamic Fuzzy Rule Interpolation

Nitin Naik*[1] | Changjing Shang[2] | Paul Jenkins[1] | Qiang Shen[2]

[1]Defence School of Communications and Information Systems, Ministry of Defence, United Kingdom
[2]Department of Computer Science, Aberystwyth University, United Kingdom

Correspondence
*Nitin Naik, Defence School of Communications and Information Systems, Ministry of Defence, United Kingdom. Email: nitin.naik100@mod.gov.uk

Abstract

Dynamic fuzzy rule interpolation (D-FRI) technique delivers a dynamic rule base through the utilisation of fuzzy rule interpolation to infer more accurate results for a given application problem. D-FRI offered dynamic rule base is very useful in security areas where network conditions are always volatile and require the most updated rule base. A honeypot is a vital part of any security infrastructure for directly investigating attacks and attackers in real-time to strengthen the overall security of the network. However, a honeypot as a concealed system can only function successfully while its identity is not revealed to any attackers. Attackers always attempt to uncover such honeypots for avoiding any trap and strengthening their attacks. Active fingerprinting attack is used to detect these honeypots by injecting purposefully designed traffic to a network. Such an attack can be prevented by controlling the traffic but this will make honeypot unusable system if its interaction with the outside world is limited. Alternatively, it is practically more useful if this fingerprinting attack is detected in real-time to manage its immediate consequences and preventing the honeypot. This paper offers an approach to building a cognizant honeypot for detecting active fingerprinting attacks through the utilisation of the established D-FRI technique. It is based on the use of just a sparse rule base while remaining capable of detecting active fingerprinting attacks when the system does not find any matching rules. Also, it learns from current network conditions and offers a dynamic rule base to facilitate more accurate and efficient detection.

KEYWORDS:
Honeypot; dynamic fuzzy rule interpolation; *D-FRI*; fuzzy rule interpolation; *FRI*; fuzzy inference system; active fingerprinting attck

1 | INTRODUCTION

Fuzzy rule-based inference offers an established mechanism for inferring outcomes given certain observations in the development of many intelligent systems. Fuzzy inference is very similar to human reasoning but their effectiveness is profoundly dependent on the fuzzy rule base and its specification. The conventional fuzzy inference system (FIS) is employed to infer outcomes based on any matching rules in the given rule base. Nonetheless, it is only effective when the rule base covers entire problem domain (i.e., no outcome is out of the rule base). Sometimes it is not possible to achieve such a complete rule base and when it is achievable can cause other issues such as significant computational overheads and possible redundancy and conflict. In situations where the rule base is incomplete, outcomes can be successfully drawn using fuzzy rule interpolation (FRI). However, both the complete or incomplete rule bases are generally static in nature and may become ineffective in long run if not update

[0]**Abbreviations:** D-FRI, dynamic fuzzy rule interpolation; FRI, fuzzy rule interpolation; FIS, fuzzy inference system

over time. Dynamic fuzzy rule interpolation (D-FRI) technique is particularly developed to solve this problem, which offers the most updated rule base for a given application through the utilisation of FRI (Naik, Diao, & Shen 2014). D-FRI technique can be employed for any fuzzy intelligent system irrespective of its type of rule base.

The employment of a dynamic rule base derived from D-FRI offers a range of real benefits for security applications to include the concurrent traffic conditions for inference. It has been successfully exploited in several security areas such as D-FRI-Snort (Naik, Diao, & Shen 2016), D-FRI-WinFirewall (Naik, Diao, Shang, Shen, & Jenkins 2017), ID-Honeypot (Naik, Shang, Shen, & Jenkins 2018a) and VD-Honeypot (Naik, Shang, Shen, & Jenkins 2018b). In this work, the utilisation of D-FRI is extended to build a cognizant honeypot for detecting active fingerprinting attacks. This utilisation of D-FRI with a honeypot is aimed at preventing a honeypot from being identified by attackers because a honeypot is only useful when it is concealed. Once it is identified by attackers, it can be compromised and then led to attacking on others (Grimes 2017; Spitzner 2003). The employment of the D-FRI technique enables honeypot to detect and predict active fingerprinting attacks and thus, the subsequent action can be taken to prevent the honeypot.

To investigate and resolve the above problem through the use of D-FRI, this experimentation-based investigation employs the KFSensor honeypot and Nmap fingerprinting tool. There are five specific active fingerprinting attacks performed on the honeypot using Nmap. Subsequently, attack simulation data is collected and analysed to detect the sign of an active fingerprinting attack. The most influential attributes from the data are extracted and utilised to design a fuzzy inference system in integration with D-FRI for this cognizant honeypot. The D-FRI based honeypot can detect active fingerprinting attacks utilising the sparse rule base under both conditions when it finds or does not find any matching rule(s). Additionally, it provides the most updated rule base to avoid interpolation overheads as much as possible and at the same time, also to enable the production of the outcome even if no matching rule is found. Thus, this work makes the honeypot a cognizant honeypot for detecting active fingerprinting attacks.

This paper is structured into the following sections: Section 2 explains the working procedure of D-FRI, honeypot and its types, and the concept of active fingerprinting attack. Section 3 discusses the simulation of active fingerprinting attacks to collect the attack data. Section 4 identifies the most influential attributes to detect an active fingerprinting attack on the honeypot. Section 5 presents the development of the cognizant honeypot through the utilisation of D-FRI. Section 6 demonstrates the experimental results of the cognizant honeypot against the simulated active fingerprinting attacks. Section 7 discusses the main limitations of the proposed approach. Section 8 summarises this work and considers further enhancements.

2 | BACKGROUND

2.1 | Dynamic Fuzzy Rule Interpolation (D-FRI)

A conventional fuzzy inference system infers outcomes using dense rule base that covers the complete problem domain. When it is not possible to cover the complete problem domain then a sparse rule base is resorted to and the most effective way to draw outcomes is the utilisation of an FRI system. Inference and interpolation can be combined together for designing more effective fuzzy inference systems based on the sparse rule base. This integrated system can offer several benefits including where feasible, performing conventional inference with the sparse rule base and minimising interpolation overheads. Nonetheless, the effectiveness of such an integrated system may be affected due to the static nature of the sparse rule base. This observation inspires the development of an additional mechanism for intelligent learning and adaptation of the original rule base.

Techniques for dynamic fuzzy rule interpolation (D-FRI) have been particularly proposed to deal with this important issue. They are capable of offering the most updated rule base through the utilisation of FRI (Naik et al. 2014; Naik, Diao, & Shen 2018). The underlying mechanism can also be used to gradually develop a dense rule base from an original sparse rule base if needed. The D-FRI system is the result of an integration of fuzzy inference, fuzzy rule interpolation, and dynamic rule learning and adaptation techniques. It can be employed to resolve a wide range of problems and one of which is cyber security, where the changing network traffic conditions require continuous modification of a rule base (Naik et al. 2017). The D-FRI system offers a generic framework to encompass any fuzzy inference and interpolation technique, however, in this implementation, the Mamdani's fuzzy inference (Mamdani & Assilina 1975) and transformation-based rule interpolation (T-FRI) (Huang & Shen 2006; Yang, Chao, & Shen 2017) are utilised.

The working of D-FRI is illustrated in Figure 1. It starts with attempting to match a newly presented observation against a set of original rules $\mathbb{R}$, which is sparse (or incomplete) in nature. Subsequently, an FRI system (here T-FRI) is utilised to perform interpolation in the absence of any matched rule that generates a pool of interpolated rules $\mathbb{R}'$ over a particular time period. The domains of antecedents of $\mathbb{R}'$ are divided into a set of hyper-cubes $\mathbb{H}$. Then, all the non-empty blocks $\mathbb{H}^*$ are selected to perform GA-based clustering, which determines the "best" clustering arrangement leading to a set of strong hyper-cubes $\mathbb{H}^1$ and another of weak hyper-cubes $\mathbb{H}^0$. The weak hyper-cubes are less concentrated with very few rules and hence, are merged into certain strong hyper-cubes to obtain the final clusters. This GA-based clustering algorithm does not

require to set the number of clusters k in advance in an effort to produce the best clustering arrangement. For generating dynamic rules, the best clusters are selected based on the empirically set threshold σ of a number of interpolated rules. These selected rule clusters are transformed into new rules utilising an aggregation method and subsequently, promoted to the rule base $\mathbb{R}$. This D-FRI is a flexible approach and any inference and interpolation technique may be employed to generate the dynamic rules. It can help to reduce interpolation overheads for the otherwise repetitive interpolation processes through direct rule inference with a promoted rule that matches the observation afterwards.

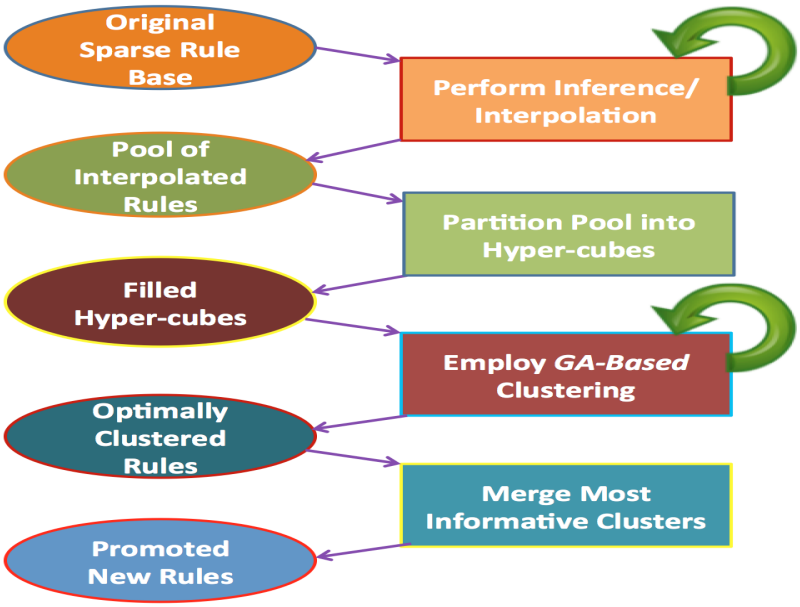

FIGURE 1 Dynamic Fuzzy Rule Interpolation (D-FRI), Naik et al. (2014)

2.2 | Honeypots

A honeypot is a concealed system to trick attackers in believing that they are dealing with a real system for collecting real-time information about the potential attacks and attackers (Grimes 2017). Such real-time information helps security experts learn vulnerabilities of their organisational network and hence, to encourage them to develop a more robust security policy and safe network (Rowe 2006). Based on the activity and capability of a honeypot, it may be referred to as a low-interaction honeypot or a high-interaction honeypot. Low-interaction honeypots are those which are designed with limited resources and capability to deal with attackers. They only emulate services and Operating Systems (OS's) to lure attackers (Spitzner 2003). High-interaction honeypots are those which are designed with full resources and capability to deal with attackers. They offer real services and OS's to lure attackers (Spitzner 2003). Despite honeypots are very useful in collecting real-time information about attacks and attackers, they should not be employed as an alternative to replace other security systems such as firewall and IDS/IPS (Naik & Jenkins 2018b; Naik, Jenkins, Cooke, & Yang 2018).

2.3 | Fingerprinting

In general, fingerprinting is a method used to identify any targeted network or system through collecting typical information from that system (Naik, Jenkins, Cooke, & Yang 2018). Fingerprinting can be a normal operation or an attack depending on the authority and motive of the individual who is performing it. If an unauthorized operation is carried out for the purpose of exploiting the information for further attacks to the target then it is termed as a fingerprinting attack (Naik, Jenkins, & Savage 2018). Fingerprinting attacks may be done either actively or passively on the target (Naik & Jenkins 2018a). In an active fingerprinting attack, a group of fabricated packets is sent to the target system and received response is analysed to obtain a fingerprin. In a passive fingerprinting attack, only the traffic of the target machine is intercepted and analysed to obtain a fingerprint.

2.4 | TCP/IP Stack or OS Fingerprinting Attack

The most common fingerprinting attack is an OS or TCP/IP stack fingerprinting attack, which is aimed at obtaining the fingerprint of an OS and other related information of the target machine. In an active OS fingerprinting attack, a group of fabricated packets is sent to the target system for obtaining a reply comprising fingerprint related information (Allen 2008). The reason for the success of this type of OS fingerprinting attack is that the same TCP/IP protocol stack is implemented by different OS's differently. As a result of this, every OS generates a unique TCP/IP response for the same incoming TCP/IP traffic, which is utilised to identify the information about an OS of the target machine.

3 | EXPERIMENTAL SIMULATION: COLLECTING FINGERPRINTING ATTACK DATA

This work takes an experimentation-based approach, where only active OS or TCP/IP fingerprinting attacks are carried out on the honeypot to detect the details of their OS and hardware device. This information may help attacker in identifying the honeypot system through their actual OS or any ambiguity in simulated OS. Upon identification, honeypot may be compromised and exploited to attack others. The simulation of active fingerprinting attacks is performed on the KFSensor honeypot using the Nmap fingerprinting tool. KFSensor honeypot is a low-interaction one, which is designed for Windows platform (Keyfocus.net 2018). It is easy to configure and use through its graphical user-interface. Nonetheless, similar experiments can be performed on any other honeypot to collect the attack data using the same Nmap fingerprinting tool. Based on the analysis of collected attack simulation data, the most influential attributes from the data will be extracted as signs of an active fingerprinting attack.

This active fingerprinting attack simulation is based on the assumption that the Nmap tool can perform several different types of fingerprinting attack, but the present experiments are focused on OS or TCP/IP fingerprinting attacks only, where Nmap sends a sequence of TCP/IP packets (generally 16 packets excluding retransmission packets) to the targeted open and close ports of the aimed system (Lyon 2009d). It can also retransmit several packets if it does not receive any response from the aimed system (e.g., when the packets sent to certain specific open/closed ports are unavailable to reply or the sent packets are lost (Greenwald & Thomas 2007)). All the TCP/IP packets are IPv4 packets and are generated with random IP ID values. These specially crafted packets are utilised to target some of the known issues of the RFC standards of TCP/IP protocol suite. After receiving a response from the aimed system, Nmap evaluates the values of several TCP/IP attributes to determine an OS fingerprint based on its OS fingerprinting database.

For this attack simulation, five Nmap active OS fingerprinting attack commands are chosen and tested with different switches as shown in Table 1. All these commands are used to obtain an OS fingerprinting which may include information such as OS names, its versions, device types and other architectural information (Lyon 2009e). The first fingerprinting attack command detects the OS fingerprints of the aimed system and provides a comprehensive account of each fingerprint (Lyon 2009f). The second attack command performs several operations such as OS detection, version detection, script scanning, and trace route. The third command utilises *fuzzy* logic, performing the most effective OS fingerprinting by detecting the closest matched OS in the event of no exact matched OS is found (Lyon 2009a). It shows all possible closely matched OS with their confidence level in percentage. The fourth fingerprinting attack command is slightly different; it detects OS through identifying the running services on various ports such as HTTP, SSH, FTP, Telnet, SMTP, and DNS. The fifth fingerprinting attack command performs remote OS detection for *n* times, where *n* is the level of the intensity ranging from 0 to 9, with a higher number representing a higher intensity for more precise fingerprinting results (Lyon 2009b 2009c).

TABLE 1 Selected Active Fingerprinting Attack Commands of NMAP

No.	Active Fingerprinting Attack Command	Given Name
1	nmap − O < Honeypot IP >	*Fingerprinting Attack1*
2	nmap − A < Honeypot IP >	*Fingerprinting Attack2*
3	nmap −O−fuzzy −−osscan−guess < HoneypotIP >	*Fingerprinting Attack3*
4	nmap −sV −−version−intensity n < Honeypot IP >	*Fingerprinting Attack4*
5	nmap −O −−max−OS−tries n < Honeypot IP >	*Fingerprinting Attack5*

For collecting a sufficient amount of consistent traffic patterns of these attacks, each attack with different switches and variations is run approximately100 times and thus, all five attacks together are run for nearly 500 times. The data is collected into two logs through KFSensor honeypot and Wireshark analyser (Wireshark 2019) for comparative analysis. Wireshark can capture slightly different traffic from KFSensor honeypot since this honeypot records events according to their configuration and priority and therefore, may miss some of the traffic.

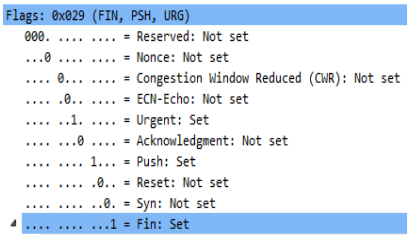

FIGURE 2 Captured TCP packet with URG/PSH/FIN scanning during an OS fingerprinting attack

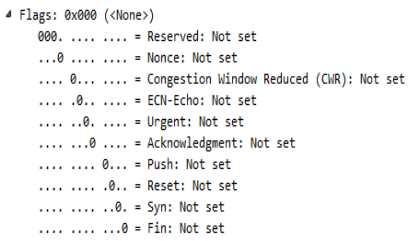

FIGURE 3 Captured TCP packet with NULL scanning during an OS fingerprinting attack

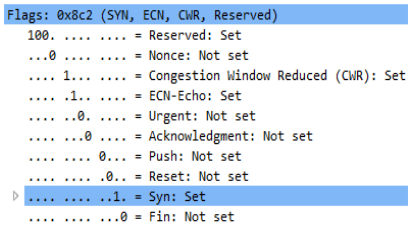

FIGURE 4 Captured TCP packet with Reserved Bit and ECN-Echo scanning during an OS fingerprinting attack

4 | DETECTING SIGNS OF FINGERPRINTING ATTACKS

In the previous section, it has been introduced how the attack simulation data is collected that will be further analysed to discover some of the influential attributes as signs of an active fingerprinting attack. This data analysis is based on underlying principles of several fingerprinting tools Nmap, Xprobe2, IPlog, Amap and Nessus. Typically, common TCP/IP protocols and procedures are employed in most of the selected tools to perform this form of attack. Based on the working similarity of fingerprinting tools, this analysis of Nmap data may help to determine certain common signs of the fingerprinting attack for most of the tools. As such an OS fingerprinting attack utilises the TCP/IP protocol stack, the main focus is to analyse signs of an active fingerprinting attack in TCP/IP (TCP, ICMP and UDP) packets. Nonetheless, some fingerprinting tools extract information from TCP packets and some from ICMP packets. This reflects that a generic detection technique should include signs from both TCP packets and ICMP packets.

4.1 | Detecting Malicious TCP Flags

TCP flags and their uncommon usages are very common in several attacks and also in an active fingerprinting attack. TCP flags are used to control or indicate the flow of data or state of connection. It comprises of six common flags (SYN, ACK, URG, PSH, RST, FIN) and two additional flags (CWR, ECN) alongside three Reserved Bits. Most of the uncommon or illegal flag(s) or group of flags are explained here, out of which several flags are found in the simulated attack data and can be used as one of the sign of an active fingerprinting attack.

4.1.1 | FIN Scanning

This flag implies the end of communication during the TCP session. The use of a single FIN packet is uncommon and it is not anticipated without an earlier established connection. In fingerprinting attack, an attacker can utilise this FIN packet to obtain response from the aimed system. If the targeted port is open then this FIN packet will be ignored by many OSs, otherwise, some kind of reply (e.g., RST packet) may be sent to the attacker. Windows OS replies with an FIN packet which can be a sign of this OS on the aimed system. The main advantage of using this FIN scanning is that it may not be detected as a suspicious packet by common firewalls, filters or scanners.

4.1.2 | FIN/SYN Scanning

The purpose of these two flags FIN and SYN are quite different and generally not set in the same TCP packet. An attacker can utilise this FIN/SYN packet to obtain response from the aimed system. Depending on the response, it can determine the type of an OS such as many Linux versions reply with a FIN/SYN/ACK packet.

4.1.3 | URG/PSH/FIN Scanning

This group of flags is one of the most popular abnormal flag group called Xmas scan. This utilises an ambiguity in RFC 793 of TCP for learning about open and closed ports at the aimed system. An attacker can utilise this URG/PSH/FIN packet to obtain response from the aimed system. If the targeted port is open then this URG/PSH/FIN packet will be ignored by many OS's, otherwise, some kind of reply (e.g., RST/ACK packet) may be sent to the attacker. However, certain OS's only send RST packets as a reply irrespective of whether the port is open or not. In any case, the list of OS's and their behaviours are well known and it is easy to determine the type of an OS. However, this URG/PSH/FIN scanning is only effective

for those OS's which conform to RFC 793. Figure 2 shows a captured TCP packet with URG/PSH/FIN scanning during an active OS fingerprinting attack.

4.1.4 | NULL Scanning

It is very similar to URG/PSH/FIN scanning and also, generates almost similar responses. In this NULL scanning, a number of specially crafted TCP packets with no flag set but with a sequence number are sent to the aimed system. An attacker can utilise this NULL packet to obtain response from the target. If the targeted port is open then this NULL packet will be ignored by many OS's, otherwise, some kind of reply (e.g., RST packet) may be sent to the attacker. Figure 3 shows a captured TCP packet with NULL scanning during an active OS fingerprinting attack.

4.1.5 | Reserved Bit Scanning

Alongside TCP flag, there are three reserved bits for the future. Generally, these three bits are set to zero in all TCP packets by default. An attacker can utilise this reserved bits and set in TCP packet to obtain a specific response from the aimed system. Figure 4 shows a captured TCP packet with Reserved Bit scanning during an active OS fingerprinting attack.

4.1.6 | ECN-Echo Scanning

This is an extended flag of TCP called Explicit Congestion Notification (ECN). It is used to notify about network congestion and it does not drop packets. This ECN can be used only between the two ECN-enabled endpoints and it is an optional feature. An attacker can utilise this flag bit and set in TCP packet to obtain a specific response from the aimed system. Figure 4 shows a captured TCP packet with ECN-Echo scanning during an active OS fingerprinting attack.

4.2 | Detecting Malicious TCP Options

TCP Options is another important attribute of TCP packet that can be employed in an active fingerprinting attack due to the flexibility of its use and variable size (0 to 40 bytes). Every OS follows a particular pattern for its TCP Options which can be used as another sign of an active fingerprinting attack.

4.3 | Detecting Malicious ICMP Requests

A number of fingerprinting tools utilise ICMP packets to obtain an OS fingerprint of the aimed system. ICMP is most commonly used for network management and troubleshooting purposes such as to check whether the target is alive or not. An attacker can utilise ICMP packets to obtain a specific response from the aimed system to know the OS of this system. Some commonly used crafted ICMP requests for fingerprinting attacks are ICMP Echo Request (Type 8), ICMP Router Solicitation Request (Type 10), ICMP Timestamp Request (Types 13), ICMP Information Request (Type 15 -Deprecated) and ICMP Address Mask Request (Type 17 -Deprecated). Their purpose is to collect significant information about the OS of the target machine. These ICMP requests are different from the normal ICMP requests because of several ambiguties such as no payload or invalid values regarding certain options of the ICMP header.

4.4 | Detecting Malicious UDP Requests

UDP packets are normally used in conjunction with TCP and ICMP packets in an active fingerprinting attack to collate the complete and accurate fingerprint of the aimed system. These packets are sent at the start of communication to discover several open and closed ports for further exploitation. UDP is quite convenient protocol as it does not require any connection, therefore, UDP packet can be sent directly to the aimed system to obtain quick response. An attacker can utilise UDP packets to obtain a specific response from the aimed system to know the OS of this system. If the targeted UDP port is open then this packet will be ignored by many OS's, otherwise, some kind of reply (e.g., an ICMP error message - *Destination Unreachable*) may be sent to the attacker. The set DF ("Don't Fragment") bit of a UDP packet can also cause an aimed system to send an ICMP error message in some cases.

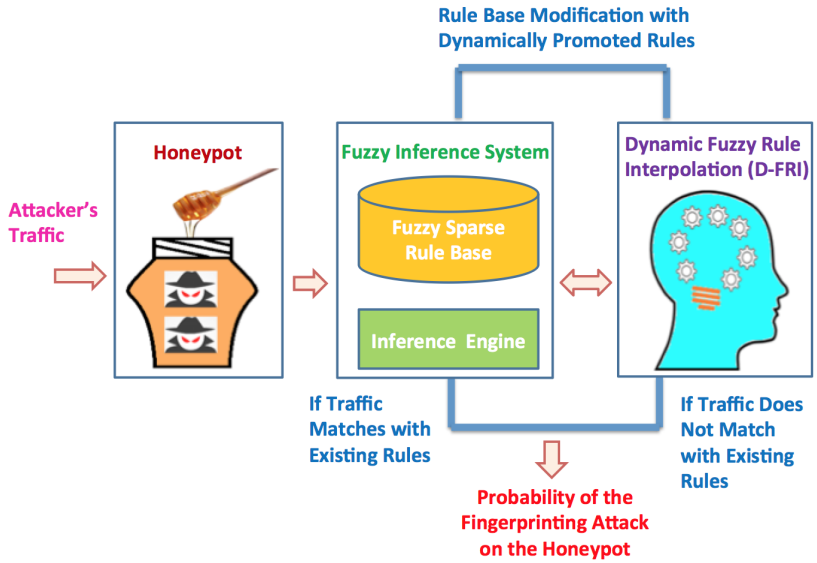

FIGURE 5 Functional diagram of cognizant honeypot for detection of active fingerprinting attacks

5 | BUILDING A COGNIZANT HONEYPOT FOR DETECTING ACTIVE FINGERPRINTING ATTACKS

The design of the present system is focused on low-interaction honeypots which are usually devised with limited resources and capabilities. Therefore, it is important to prioritise the targets and incidents for these honeypots while ignoring certain lower priority targets and incidents. In case of active fingerprinting attacks, honeypots may be able to record all related activities but it may not be possible to link up all these activities to the determination of this attack being a particular active fingerprinting attack and not any other attack. To supply such intense detection ability further computational intelligence is integrated with the honeypot, enabled by the use of D-FRI. As a result, the implemented system herein has the focussed ability of detecting and predicting all those active fingerprinting attacks outlined in the preceding section. The procedural diagram of this proposed cognizant honeypot is illustrated in Figure 5. It contains three main components: any low-interaction honeypot, a fuzzy inference system and a D-FRI system.

5.1 | Fuzzy Inference System

The generation of the original sparse fuzzy rule base is a combinational process of utilising expertise and data, where domain experts perform a reverse engineering process on the fingerprinting attack simulation data collected over a number of days. Initially, the experts identify those attributes which simulated attackers utilise to extract fingerprinting information of a victim system/network. However, most of these attributes may be good for the attackers but do not help to identify fingerprinting attacks. This observation has led to a more intensive examination of fingerprinting data logs and symptoms to establish the links between crucial attributes and a given fingerprinting attack. Expertise is also important to select those attributes which are generic and can be exploited to identify fingerprinting attacks carried out by those other than the specific fingerprinting attack tool used in the experiments. Four previously analysed attributes are so identified as signs of active fingerprinting attacks: Malicious TCP Flags, Malicious TCP Options, Malicious ICMP Requests and Malicious UDP Requests. Their value ranges are analysed meticulously by the experts and fed to the Matlab Fuzzy Logic Designer tool box to generate the fuzzy rules. Nonetheless, this work is a preliminary investigation; in future, more intelligible fuzzy or soft computing methods (e.g., (Chen, Shang, Su, & Shen 2018)) can be utilised to generate the original sparse rule base.

This D-FRI-based system utilises both inference and interpolation with a sparse rule base, offering flexibility for an appropriate reasoning mechanism to be chosen for use, depending on the given observation. In designing the original sparse rule base and fuzzy inference system, four fuzzy input variables are derived from the four attributes identified as signs of an active fingerprinting attack as addressed in the previous section. These fuzzy input variables are called Malicious TCP Flags (MTCPF), Malicious TCP Options (MTCPO), Malicious ICMP Requests (MICMPR) and Malicious UDP Requests (MUDPR). The value range for all these four input variables are determined as 1-15 packets based on the experimental analysis of the collected attack data. Furthermore, five fuzzy sets are derived for each of these input variables, namely: Very Low, Low, Medium, High and Very High, reflecting the five severity levels of an active fingerprinting attack respectively. The underlying value range or support is defined for these

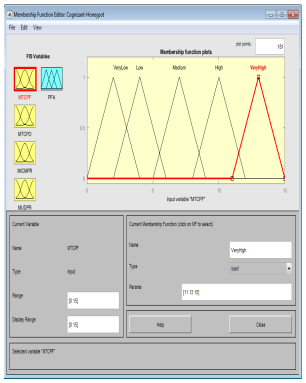

FIGURE 6 Fuzzy input variable MTCPF and its value domain

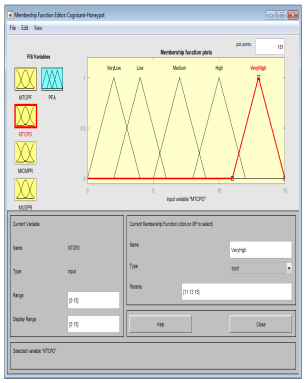

FIGURE 7 Fuzzy input variable MTCPO and its value domain

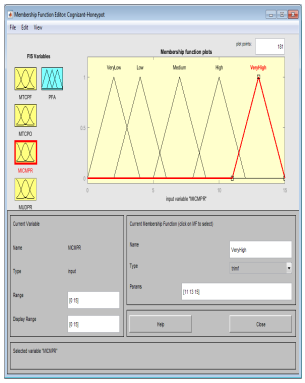

FIGURE 8 Fuzzy input variable MICMPR and its value domain

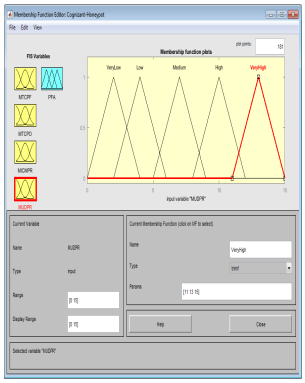

FIGURE 9 Fuzzy input variable MUDPR and its value domain

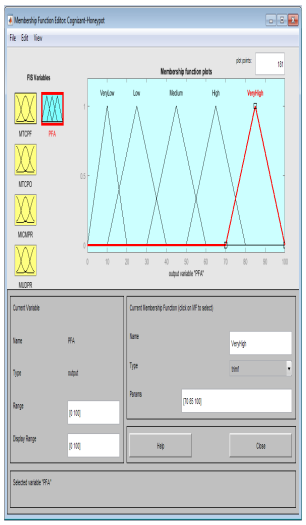

FIGURE 10 Fuzzy output variable PFT and its value domain

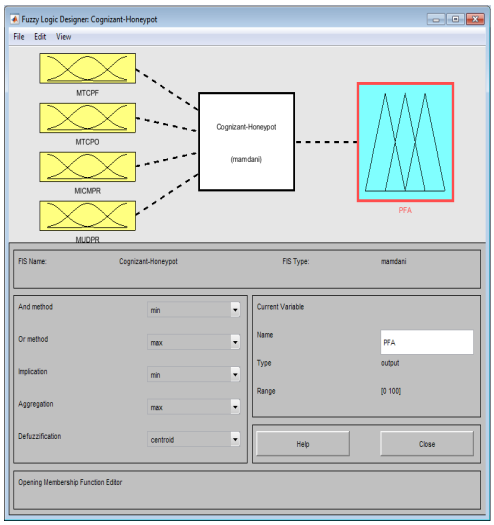

FIGURE 11 Fuzzy rule-based inference in intelligent dynamic honeypot

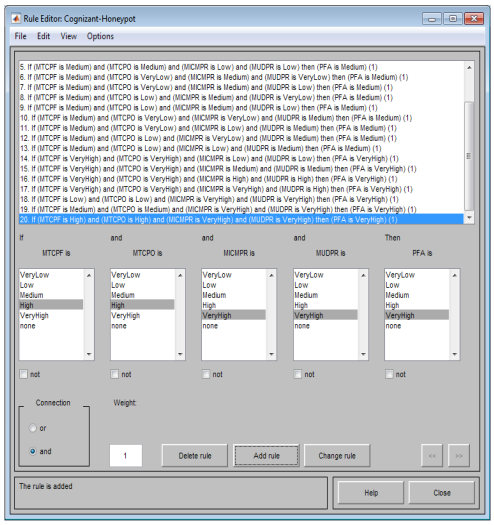

FIGURE 12 Initial sparse rule base on low-interaction honeypot

fuzzy set as follows: Very Low is 0-4 packets, Low is 2-6 packets, Medium is 5-9 packets, High is 8-12 packets, and Very High is 11-15 packets. This design process is illustrated in Figures 6 to 9, in which, each fuzzy set is chosen as a triangular membership function for this initial implementation.

For predicting the possibility of an active fingerprinting attack through the fuzzy inference system, the fuzzy output variable called Probability of Fingerprinting Attack (PFA) is derived for the original sparse rule base. Its value range is between 0-100%, and it is also split into similar five fuzzy sets: Very Low, Low, Medium, High and Very High. The support value range is specified for these fuzzy set as follows: Very Low is 0-20%, Low is 10-40%, Medium is 30-60%, High is 50-80% and Very High is 70-100%. This design specification is illustrated in Figure 10, in which, for simplicity, each fuzzy set is defined as of a triangular membership function for this initial implementation.

These fuzzy input and output variables are utilised to design the original sparse rule base and fuzzy inference system (Mamdani & Assilina 1975), which is illustrated in Figure 11. The designed original rule base is sparse, including one a small number of rules whose outcomes are either very low, medium or very high, as illustrated in Figure 12. This basic fuzzy inference system can only detect those active fingerprinting attacks which can be matched with any existing rule in this sparse rule base.

5.2 | D-FRI System for Cognizant Honeypot

Apart from the limitation of sparse rule based detection through the use of just conventional fuzzy inference system, the low-interaction honeypots are limited in their resources and capabilities, which may also affect the detection performance. The system may not collect all traffic information,

and its operation is affected due to perpetual change in network conditions. Yet, the fuzzy inference system is only effective when the input conditions are met at least partially with any existing rule, otherwise, it will not be able to generate any output resulting in no detection. To minimise this shortcoming, the D-FRI system is employed here.

D-FRI addresses the aforementioned problem, avoiding any situation where no detection is returned when there is no match is available in the rule base. Initially, it performs a linear interpolation of the selected fuzzy rules which are close to but not matching a given observation, generating an outcome and storing every such interpolation result, termed interpolated rule, for subsequent dynamic generalisation. After reaching a pre-specified threshold of the number of interpolated rules, dynamic learning is applied on them in order to obtain and promote the most concurrent rules to the sparse rule base for future use. This dynamic rule promotion procedure enhances the overall inference process in two ways: improving the efficiency by increasing the possibility of directly applying fuzzy inference (thus reducing interpolation overheads) in the future, and strengthening the effectiveness by introducing more accurate and concurrent rules. Overall, this D-FRI system reinforces the detection rate of an active fingerprinting attack, based on the recent traffic and updated rules. This will be empirically shown in the experimental section below.

6 | EXPERIMENTAL RESULTS

The detection capability of the cognizant honeypot developed in this work is tested for the previously explained five fingerprinting attacks of Nmap (as introduced in Section 3). This experimental study runs on a sparse fuzzy rule base, which is typically designed to cover only very low, medium and very high active fingerprinting attacks; the rule base does not cover any of the low and high active fingerprinting attacks. Three different experimentations are performed for the cognizant honeypot to evaluate its detection efficiency by employing the fuzzy inference system, the standard transformation-based FRI system and the D-FRI system respectively. A total of 50 active fingerprinting attacks are simulated including 10 for each type of attack in every experiment.

6.1 | Experimental Results Using Fuzzy Inference System Alone

The first experimental evaluation of the cognizant honeypot is through the utilisation of just the fuzzy inference system, where the selected five types of active fingerprinting attack are carried out on the honeypot. The detection results of this FIS-based attack simulation are listed in Table 2, where, 38 active fingerprinting attacks are detected from a total of 50 attack simulations (i.e., 76%). In particular, 31 active attacks are detected as a very high attack and 7 active attacks as a medium attack. Nonetheless, the system fails to detect 12 active attacks due to the sparsity of the rule base (i.e., 24%). It could have detected these 12 attacks had the employed rule base been devised to be a completely dense rule base, but that would require a 5 times larger rule base than the current sparse rule base. In many cases, designing or learning from data consistent dense rule bases is a challenging task, even for domain experts. Irrespective of the design challenges, every missed attack affects the detection accuracy of the honeypot. Instead of expanding the rule base, considering an alternative approach to avoid this requirement leads to the utilisation of the FRI system and D-FRI system, as evident below.

TABLE 2 Detection Results of Active Fingerprinting Attacks on Honeypot Using Original Sparse Rule Base and Fuzzy Inference System

No.	Active Fingerprinting Attack	Prediction of Active Fingerprinting Attack for 10 Simulations per Attack Type				
		Very Low	Low	Medium	High	Very High
1	*Fingerprinting Attack1*	0	0	0	0	7
2	*Fingerprinting Attack2*	0	0	0	0	10
3	*Fingerprinting Attack3*	0	0	0	0	7
4	*Fingerprinting Attack4*	0	0	7	0	0
5	*Fingerprinting Attack5*	0	0	0	0	7

6.2 | Experimental Results Using Standard FRI System

The second experimental evaluation of the cognizant honeypot is through the utilisation of the standard transformation-based FRI (T-FRI) system, where the selected five types of active fingerprinting attacks are carried out on the honeypot. The detection results of this FRI-based attack

simulation is shown in Table 3, where, all the 50 active fingerprinting attacks are detected (10 per each type of attack), however, with different severity levels. Forty active fingerprinting attacks (i.e., 80%) are detected with the higher severity level (34 of a very high and 6 of a high level), 8 active fingerprinting attacks are detected with a medium level, and the remaining 2 attacks are detected with a low level. The employment of FRI has improved the detection accuracy of the cognizant honeypot, although it has still regards 10 active fingerprinting attacks as below the higher severity level. Therefore, this cognizant honeypot is further tested through the utilisation of the D-FRI system as follows.

TABLE 3 Detection Results of Active Fingerprinting Attacks on Honeypot Using Original Sparse Rule Base and T-FRI System

No.	Active Fingerprinting Attack	Prediction of Active Fingerprinting Attack for 10 Simulations per Attack Type				
		Very Low	Low	Medium	High	Very High
1	*Fingerprinting Attack1*	0	0	1	1	8
2	*Fingerprinting Attack2*	0	0	0	0	10
3	*Fingerprinting Attack3*	0	0	0	2	8
4	*Fingerprinting Attack4*	0	2	7	1	0
5	*Fingerprinting Attack5*	0	0	0	2	8

6.3 | Experimental Results Using D-FRI System

The third experimental evaluation of the cognizant honeypot is through the utilisation of the D-FRI system, where the selected five types of active fingerprinting attack are carried out on the honeypot. The detection results of this D-FRI-based attack simulation are given in Table 4, where, similar to the FRI-based simulation outcomes, all 50 active fingerprinting attacks are detected. Although the detections are still associated with different severity levels, this implementation detects 42 active fingerprinting attacks (i.e., 84%) with the higher severity level (34 of a very high and 8 of a high level), 7 active fingerprinting attacks are identified as of a medium level and only 1 with a low level. As such, the employment of the D-FRI system has led to better results in comparison to the use of the FRI-based simulation while using the identical initial sparse rule base.

TABLE 4 Detection Results of Active Fingerprinting Attacks on Honeypot Using Original Sparse Rule Base and D-FRI System

No.	Active Fingerprinting Attack	Prediction of Active Fingerprinting Attack for 10 Simulations per Attack Type				
		Very Low	Low	Medium	High	Very High
1	*Fingerprinting Attack1*	0	0	0	2	8
2	*Fingerprinting Attack2*	0	0	0	0	10
3	*Fingerprinting Attack3*	0	0	0	2	8
4	*Fingerprinting Attack4*	0	1	7	2	0
5	*Fingerprinting Attack5*	0	0	0	2	8

In both FRI and D-FRI based experimentations, the active *Fingerprinting Attack4* was the only attack which was detected mostly below the higher severity level. This is due to the reason that this type of attack employs a different attack procedure and thus generates slightly different traffic, which affects the prediction level. Indeed, the active *Fingerprinting Attack4* uses a distinct nmap database - *nmap-services* (Lyon 2011), different from the other four types of attack which use database - *nmap-os-db* (Lyon 2017). It detects the running services for determining the operating system on the target machine (e.g., the detection of Microsoft Exchange service indicates a strong possibility of Windows OS running on that machine (Lyon 2009c)). Notwithstanding, despite that the active *Fingerprinting Attack4* type is so distinct from the rest, the D-FRI system can work better results than using the standard T-FRI system alone.

6.4 | Accuracy of Dynamic Rules

As discussed previously, the original sparse rule base is designed to cover only very low, medium and very high active fingerprinting attacks. As such, it misses other two categories, low and high active fingerprinting attacks. Having carried out hundreds of attack simulations concerning these

two categories, hundreds of interpolated rules are collated and processed through the D-FRI clustering approach, resulting in 20 clusters that are above the set rule threshold (σ=15), covering 400 interpolated rules all together. From such a dynamic learning process, only these 20 clusters are converted to 20 new rules with the D-FRI aggregation method which are subsequently promoted into the original rule base.

These new rules are added to the sparse rule base to enhance the honeypot's ability to generate correct results and minimise both future computational effort and future interpolation errors, when these rules are directly matched against any new observation. The accuracy of these dynamic rules is compared against the direct use of interpolated rules ($\epsilon_{\%\text{dvi}}$) and the *ground-truth* rules ($\epsilon_{\%\text{dvt}}$) which are generated on the basis of translating the underlying defining fuzzy grids, in exactly the same way as used to create those original rules in the sparse rule base. The differences between the use of just static rule interpolation and the *ground-truth* rules ($e_{\%\text{ivt}}$) are also provided.

In all comparisons, the percentage error $\epsilon_{\%} = \epsilon/\text{range}_y$ is computed with respect to the underlying range of the consequent variable. Table 5 exhibits the average values and standard deviations regarding the above error measures. The results clearly support the present work, by promoting the more accurate rules through the use of D-FRI than just the use of static rule interpolation, as the results of the intelligent dynamic honeypot are closer to the use of the *ground-truth* rules. Importantly, this also avoids a significant amount of computation for the otherwise required rule interpolations.

TABLE 5 Accuracy of Dynamic Rules

Metric	$\epsilon_{\%\text{dvi}}$	$\epsilon_{\%\text{dvt}}$	$\epsilon_{\%\text{ivt}}$
AVG	2.48	**1.39**	2.64
SD	2.77	**1.40**	2.74

7 | LIMITATIONS OF PROPOSED APPROACH FOR FINGERPRINTING ATTACKS

The proposed technique is promising in identifying fingerprinting attacks on honeypots, however, it has several limitations:

- It is derived using the attack method of the most popular fingerprinting tool Nmap, therefore, it may not generate accurate results for other fingerprinting tools.
- It is based on the main TCP/IP protocols TCP, UDP and ICMP, therefore, any fingerprinting attack utilising other protocols such as HTTP, SNMP may adversely affect prediction results.
- It is designed for the low-interaction honeypots because they perform limited functionalities due to the limited resources and capability, therefore, it may not prove effective for any high-interaction honeypots.
- It may generate false positives for those attacks which exhibit similar abnormalities or patterns to those particularly identified in this implementation.
- It produces fuzzy-valued indicative results only, therefore, it requires further investigation for binary interpretations of the detection outcomes should that be specifically required.
- It is dependent on the amount of traffic processed, therefore, the accuracy of the prediction is also dependent upon the amount of traffic, which may be affected by several types of network activities.

8 | CONCLUSION

This paper has presented an experimentation-based approach for the development of cognizant honeypots to detect active fingerprinting attacks on a honeypot through the utilisation of the D-FRI technique. The active fingerprinting attacks are carried out on the KFSensor honeypot using the Nmap fingerprinting tool. Having collected the attack simulation data, the most influential attributes from the data are extracted as signs of an active fingerprinting attack. Subsequently, these attributes are utilised to devise the fuzzy inference system and D-FRI system to implement the cognizant

honeypot. Experimental results have demonstrated that theD-FRI based honeypot can detect most of the Nmap-based active fingerprinting attacks utilising a sparse rule base.

While this experimental approach has only been verified on the basis of the Nmap fingerprinting tool, it is important to test whether the underlying approach would suit some other fingerprinting tools. In future, in addition to addressing those identified limitations as listed in Section 7, it is interesting to generate the sparse rule base of the proposed cognizant honeypot using data-driven techniques (e.g., methods as per (J. Li, Yang, Qu, & Sexton 2017; Tan et al. 2016)). Furthermore, other improvements may include: the use of feature selection tools (e.g., the nature-inspired (Diao & Shen 2015) or approximation-based tools (Jensen & Shen 2009)) for automated selection of the most influential attributes; the use of clustering techniques (e.g., the data reliability-based (Boongoen, Shang, Natthakan Iam-Onn, & Shen 2011) and link-based clustering (Boongoen, Shen, & Price 2010)) for self-learning of the fuzzy membership functions; and the use of alternative fuzzy rule interpolation mechanisms (e.g., the variable-ranking supported (F. Li, Shang, Li, Yang, & Shen 2018) and backward rule interpolation (Jin, Diao, Quek, & Shen 2014)) for providing more accurate interpolations.

References

Allen, J. M. (2008). *OS and Application Fingerprinting Techniques*. Retrieved 2018-01-01, from `https://www.sans.org/reading-room/whitepapers/authentication/os-application-fingerprinting-techniques-32923`

Boongoen, T., Shang, C., Natthakan Iam-Onn, N., & Shen, Q. (2011). Extending data reliability measure to a filter approach for soft subspace clustering. *IIEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, *41*(6), 1705–17143.

Boongoen, T., Shen, Q., & Price, C. (2010). Disclosing false identity through hybrid link analysis. *IArtificial Intelligence and Law*, *18*(1), 77–102.

Chen, T., Shang, C., Su, P., & Shen, Q. (2018). Induction of accurate and interpretable fuzzy rules from preliminary crisp representation. *Knowledge-Based Systems*, *146*, 152–166.

Diao, R., & Shen, Q. (2015). Nature inspired feature selection meta-heuristics. *Artificial Intelligence Review*, *44*(3), 311–340.

Greenwald, L. G., & Thomas, T. J. (2007). Toward undetected operating system fingerprinting. *WOOT*, *7*, 1–10.

Grimes, R. A. (2017). *Honeypots, in Hacking the Hacker: Learn from the experts who take down hackers*. John Wiley & Sons, Inc., Indianapolis, Indiana. Retrieved from `doi:10.1002/9781119396260.ch19`

Huang, Z., & Shen, Q. (2006). Fuzzy interpolative reasoning via scale and move transformations. *Fuzzy Systems, IEEE Transactions on*, *14*(2), 340–359.

Jensen, R., & Shen, Q. (2009). New approaches to fuzzy-rough feature selection. *Fuzzy Systems, IEEE Transactions on*, *17*(4), 824–838.

Jin, S., Diao, R., Quek, C., & Shen, Q. (2014). Backward fuzzy rule interpolation. *IEEE Transactions on Fuzzy Systems*, *22*(6), 1682–1698.

Keyfocus.net. (2018). *KFSensor: Advanced Windows honeypot system- Enhanced intrusion and insider threat detection for your network*. Retrieved 2018-01-01, from `http://www.keyfocus.net/kfsensor/`

Li, F., Shang, C., Li, Y., Yang, J., & Shen, Q. (2018). Fuzzy rule based interpolative reasoning supported by attribute ranking. *IEEE Transactions on Fuzzy Systems*, *26*(5), 2758–2773.

Li, J., Yang, L., Qu, Y., & Sexton, G. (2017). An extended takagi–sugeno–kang inference system (TSK+) with fuzzy interpolation and its rule base generation. *Soft Computing*. Retrieved from `https://doi.org/10.1007/s00500-017-2925-8`

Lyon, G. F. (2009a). *Chapter 15. Nmap Reference Guide*. Retrieved 2018-01-01, from `https://nmap.org/book/man-os-detection.html`

Lyon, G. F. (2009b). *Chapter 15. Service and Version Detection*. Retrieved 2018-01-01, from `https://nmap.org/book/man-version-detection.html`

Lyon, G. F. (2009c). *Chapter 7. Service and Application Version Detection*. Retrieved 2018-01-01, from `https://nmap.org/book/vscan.html`

Lyon, G. F. (2009d). *Chapter 8. Remote OS Detection: TCP/IP Fingerprinting Methods supported by Nmap*. Retrieved 2018-01-01, from `https://nmap.org/book/osdetect-methods.html`

Lyon, G. F. (2009e). *Chapter 8. Remote OS Detection: Usage and Examples*. Retrieved 2018-01-01, from `https://nmap.org/book/osdetect-usage.html`

Lyon, G. F. (2009f). *Nmap network scanning: The official Nmap project guide to network discovery and security scanning*. Insecure.

Lyon, G. F. (2011). *Nmap Service DB*. Retrieved 2018-01-01, from `https://svn.nmap.org/nmap/nmap-services`

Lyon, G. F. (2017). *Nmap OS Fingerprinting 2nd Generation DB*. Retrieved 2018-01-01, from `https://svn.nmap.org/nmap/nmap-os-db`

Mamdani, E. H., & Assilina, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, *7*(1), 1–13.

Naik, N., Diao, R., Shang, C., Shen, Q., & Jenkins, P. (2017). D-FRI-WinFirewall: Dynamic fuzzy rule interpolation for windows firewall. In *2017 ieee international conference on fuzzy systems (fuzz-ieee)* (p. 1-6). IEEE.

Naik, N., Diao, R., & Shen, Q. (2014). Genetic algorithm-aided dynamic fuzzy rule interpolation. In *Ieee international conference on fuzzy systems*

(pp. 2198–2205).

Naik, N., Diao, R., & Shen, Q. (2016). Application of dynamic fuzzy rule interpolation for intrusion detection: D-FRI-Snort. In *Ieee international conference on fuzzy systems* (pp. 78–85).

Naik, N., Diao, R., & Shen, Q. (2018). Dynamic fuzzy rule interpolation and its application to intrusion detection. *IEEE Transactions on Fuzzy Systems*, *26*(4), 1878-1892.

Naik, N., & Jenkins, P. (2018a). Discovering hackers by stealth: Predicting fingerprinting attacks on honeypot systems. In *2018 ieee international symposium on systems engineering (isse)*.

Naik, N., & Jenkins, P. (2018b). A fuzzy approach for detecting and defending against spoofing attacks on low interaction honeypots. In *21st international conference on information fusion (fusion)* (p. 904-910). IEEE.

Naik, N., Jenkins, P., Cooke, R., & Yang, L. (2018). Honeypots that bite back: A fuzzy technique for identifying and inhibiting fingerprinting attacks on low interaction honeypots. In *2018 ieee international conference on fuzzy systems (fuzz-ieee)*.

Naik, N., Jenkins, P., & Savage, N. (2018). Threat-aware honeypot for discovering and predicting fingerprinting attacks using principal components analysis. In *Ieee symposium series on computational intelligence (ssci)*.

Naik, N., Shang, C., Shen, Q., & Jenkins, P. (2018a). Intelligent dynamic honeypot enabled by dynamic fuzzy rule interpolation. In *The 4th ieee international conference on data science and systems (dss-2018)* (p. 1520-1527). IEEE.

Naik, N., Shang, C., Shen, Q., & Jenkins, P. (2018b). Vigilant dynamic honeypot assisted by dynamic fuzzy rule interpolation. In *Ieee symposium series on computational intelligence (ssci)*.

Rowe, N. C. (2006). Measuring the effectiveness of honeypot counter-counterdeception. In *System sciences, 2006. hicss'06. proceedings of the 39th annual hawaii international conference on* (Vol. 6, pp. 129c–129c).

Spitzner, L. (2003). *Honeypots: Tracking Hackers* (Vol. 1). Addison-Wesley Reading.

Tan, Y., Li, J., Wonders, M., Chao, F., Shum, H. P. H., & Yang, L. (2016). Towards sparse rule base generation for fuzzy rule interpolation. In *2016 ieee international conference on fuzzy systems (fuzz-ieee)* (p. 110-117).

Wireshark. (2019). *Wireshark*. Retrieved 2019-02-01, from `https://www.wireshark.org/`

Yang, L., Chao, F., & Shen, Q. (2017). Generalized adaptive fuzzy rule interpolation. *IEEE Transactions on Fuzzy Systems*, *25*(4), 839–853.

Conflicts of interest: None