Predicting Multi-Stage Attacks Based on hybrid approach

Abdulrazaq Z Almutairi, James Flint, David Parish
School of Electronic, Electrical and Systems Engineering
Loughborough University
Leicestershire, UK

Abstract

Multi-stage attacks can evolve dramatically causing much loss and damage to organisations. These attacks are frequently instigated by exploiting actions, which in isolation are legal and are therefore particularly challenging to detect. Much research has been conducted in the multi-stage detection area, in order to build a framework based on an events correlation approach. This paper proposes a framework that predicts multi-stage attacks based on a hybrid approach, which combines two techniques; IP information evaluation and process query system (PQS). This paper shows the analysis of three multi stage attacks, detailing their steps and information hitherto unexploited in current intrusion detection systems. The paper also goes through the implementation of each technique used in the hybrid approach.

1. Introduction

Multi-stage attacks have a significant impact on organisations. They have been described as the most challenging set of attacks to investigate and detect [1]. These attacks occur through multiple phases to get access to an organisation. Most of these attacks involve three phases. In the first phase, attackers try to analyse available information about the target, to find vulnerabilities and weaknesses that can be exploited. In the second phase, attackers exploit the weaknesses found in the first phase to inject malware into, or to gain access to, the system. In addition, they try to get more details and conduct a deep analysis about the system to find data or resources in which they have an interest. In the final phase, after gaining access, attackers are in a position to destroy the system or steal valuable information [2]. Different solutions have been introduced to detect multi-stage attacks, some of those being event correlation-based. Event correlation-based solutions try to match network events with certain attack patterns. When a stream of network events matches a certain pattern, attacks can be stopped before progressing to the next stages. Many researchers claim the effectiveness of that approach in detecting multi-stage attacks. However, this approach requires having prior knowledge of the multi-stage attack pattern (sequences), which is not always feasible since discovering new complex attacks normally takes some time. The Shady Rat Operation attack is a good example of that; it started in 2006 and was only discovered in 2011[3]. Thus, it has been decided to follow a different approach in this research, rather than depending only on network events correlation when proposing a solution for predicting multi-stage attacks. The proposed approach is a hybrid one based on two techniques: identity checker and event correlation. The identity checker is based on evaluating the reputation of IP addresses participating in network traffic using fuzzy logic. Fuzzy logic works on the basis of defining rules to produce an output. Based on specified rules, fuzzy logic decides whether we need to stop the traffic with evaluated IP addresses to block potential attacks. On the other hand, the event correlation component is based on using PQS.

Section 2 provides a brief background of fuzzy logic, social engineering, CRLF (carriage return line feed) injection, cross-sites scripting, and PQS. Section 3 provides an analysis of three different multi-stage attack scenarios that help in understanding the behaviour of multi-stage attacks. The first scenario is about communication with a bad DNS server and how that has been employed by an attacker to register machines to its bot army. The second scenario discusses the Shady Rat attack, which is a good example of how social engineering can be employed to target an organisation. The third scenario shows how header splitting can be employed by an attacker to target a network connected to a web host running a web application. Section 4 gives an overview of the proposed approach. Section 5 details the implementation of the identity checker. Section 6 discusses the evaluation process detailing the evaluation approach and results obtained in different evaluation phases. Section 7 discusses how PQS could be used to detect attacks. Section 8 goes through some related work. Section 9 provides the conclusion and future work based on this paper.

2. Brief Background

2.1 Fuzzy Logic

Fuzzy logic is a computational approach based on human language rules. The fuzzy systems translate the defined rules to mathematical equivalents [4]. Those systems, as shown in Figure 1, consist of a fuzzifier, inference engine, rules base, and defuzzifier. Fuzzy systems work as follows [4]:

The fuzzifier converts crisp inputs to a fuzzy set by using specified membership functions for each input.

Based on the defined rules, the inference engine produces a fuzzy output.

The fuzzy output is converted to a crisp value using the membership functions defined for defuzzifier.

Fuzzy logic is suitable for ambiguous scenarios [5], where there is no certainty about making decisions. When comparing fuzzy logic with machine learning algorithms, it has been found that constructing the fuzzy rules for a system does not take much effort and time, compared to machine learning algorithms. Machine learning algorithms require large data sets for training to obtain accurate results. In addition, the training time with a large data set is a very time consuming process [6]. However, fuzzy logic may not be suitable in scenarios where it is difficult to deduce the reasoning logic.

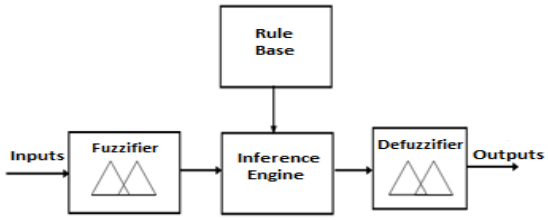

Figure 1. Fuzzy logic Components

It will be shown in section 5 that the logic of detecting multi-stage attacks can be simply modelled using 'if then' rules. Based on the nature of the problem and mentioned advantages of fuzzy logic, it will be a suitable choice for building the reasoning module in the proposed solution.

2.2 Social Engineering

Social engineering can play a role in constructing a multi-stage attack. It is the art of abusing human behaviour in order to violate security without victims realising that they have been manipulated [7].

2.3 CRLF Injection

The CRLF injection, which is also known as HTTP Response splitting, is an attack that can be easily constructed. However, it is an extremely destructive web attack. Attackers construct this kind of attack by exploiting vulnerable web applications that may also allow other types of vulnerabilities, such as cross site scripting and cross site forgery. The CRLF injection is carried out by injecting a very significant sequence of characters into web requests. This sequence contains two special characters representing EOL (end of line), which is used as a marker for many protocols, including HTTP and NNTP. In web applications, headers are split based on the position of CRLF in requests. Malicious users inject their own CRLF sequence into an HTTP request. In the absence of filtering malicious inputs, malicious users will be able to control the functionalities of a web application function. In the next section, an example of CRLF injections will be discussed, showing how CRLF injections can be employed by attackers to construct multi-stage attacks [8].

2.4 Process Query System (PQS)

PQS was defined in [9] as follows: "*a new kind of information retrieval technology in which user queries are expressed as process descriptions. The goal of a PQS is to detect the processes using a data stream or database of events that are correlated with the processes' states*". In other words, it is a software paradigm used for addressing event-processing challenges [10]. The PQS modelling framework defines processes with unique states, dynamics, and observables. In the computer security context, computer attacks are considered the processes, while stages of attacks represent states. The process dynamics are presented by network events that move the processes from one state to another one (e.g. multiple consecutive scan, irregular DNS response).

3. Multi-Stage Attack Scenarios

3.1 Scenario A

This scenario has been analysed by using a trace file that contains a capture of real network traffic [11]. The scenario gives an example of how attackers can register machines to their bot army. In this scenario, the attacker used the compromised host to contact a bad DNS server. The DNS server returned an unusual DNS response containing 11 IP addresses, while a normal response normally does not return more than five IP addresses. The attacker used the compromised host to scan IP addresses returned in the DNS query response and tried to establish communication with them. After a successful 3-handshake with one of the IP addresses returned in the response, the attacker sent packets that contained commands used by the botnet.

Some steps in this scenario could be considered to predict the occurrence of the attack. Detecting a DNS query with a bad DNS server can trigger an alert of malicious traffic. In addition, an irregular DNS

response can indicate unusual behaviour. Moreover, sending packets containing commands used by botnet gives a strong indication that the traffic is malicious.

3.2 Scenario B

One of the multi-stage attacks, that is social engineering-based, is Operation Shady Rat. This attack was categorised by MacAfee [12] as an advanced persistent threat. An Operation Shady Rat attack involves five steps. In the first step, attackers select one or more organisations, then email individuals who work at those organisations. The emails sent contain information that attracts those individuals. Those emails also contain attached files that are relevant to the email body. Those files appear to recipients as normal files such as Word, Excel, or pdf files, but they are loaded with malicious code. For example, employees in a marketing company have a high interest in getting new contacts. Therefore, attackers may target this group by sending an email attached with an Excel file containing a contacts list. In the second stage, recipients download the attached files, then open them. At the point of opening the file, the malware is installed on the victim's computer, thus compromising their computer. In the third stage, the installed malicious program tries to establish a connection with a remote site specified in the code. The remote site URL does not look suspicious and it looks like a link to an image or normal html file, but the returned contents from that URL contains some information used by the malicious code. That information cannot be seen as being suspicious content, as it appears as a part of the html content. In addition, that information may be encoded or encrypted, so it will be difficult to analyse. For example, html comments can be used to embed the information that malware uses inside the html content. The comments are visible to end users, look absolutely legitimate, and cannot be seen as any kind of threat. The html comments may contain an IP address of a remote server or a command in an encrypted or encoded format. In the fourth stage, the installed malicious code establishes a connection with the IP address obtained in the third stage. In the fifth stage, attackers at the remote site establish a remote shell and run shell commands targeting the compromised machine. Attackers at this point can upload or download from the compromised side.

All steps of this scenario look legitimate and not suspicious. However, checking the reputation of the IP addresses involved in the communication traffic between the malware code and other servers may give an indication of suspicious traffic.

3.3 Scenario C

This scenario is based on exploiting an insecure web application. An insecure web application can give a chance for attackers to get access to machines. The scenario shows how attackers exploit a vulnerable PHP web application to make a CRLF injection. The first step in this attack is carrying out a web vulnerability scan on a web server. This scan gives an attacker information about PHP configurations and different URLs, including POST and GET parameters sent with them. The attacker then uses that information to send an email to a victim containing a CRLF-manipulated link. This link looks legitimate, but it contains parameters set to values that make a vulnerable web application open a different URL, rather than the specified URL in the code. The injected URL may point to a file that runs on the victim's machine to push a remote shell for the attacker. The attacker proceeds by getting access to the web server, then downloads files or scans the network to find information they are interested in, or find targets they want to destroy.

This type of attack can be predicted or stopped at different points. The first point is checking parameters sent with web requests coming to the web server, whether it can cause CLFR injections or not. In addition to that, outgoing requests from the web server can be checked to see whether they go to trusted destinations or not.

4. The Proposed approach

The proposed approach is based on using two techniques; IP Information evaluation and process query systems as shown in Fig 2. The approach includes two components; each one is based on one of the two techniques. If the output for each component states normal traffic; the traffic will be normal. On the other hand, the traffic will be classified as malicious by this approach if one of its components classified the traffic as malicious. The idea behind using the mentioned techniques is to check the traffic from different perspectives. The first technique checks the traffic in terms of identity while the other one checks the traffic in terms of contents. Therefore, the traffic will be considered as a legal traffic only if both identity (IP participated in the traffic) and traffic contents are not malicious.

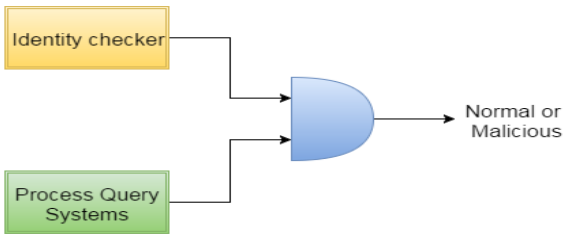

Figure 2. The proposed solution block diagram

5. The Identity checker

5.1 An overview

The identity checker is based on evaluating the reputation of IP addresses participating in the captured network traffic. The identity checker consists mainly of three modules as shown in Fig 3. The first module (Network Sniffer) is responsible for monitoring network traffic by reading incoming and outgoing traffic (It is implemented using tcpdump). This module extracts IP addresses found in network packets. The IP info finder is responsible for finding information related to the IP addresses. The information obtained by the second module includes IP geographic information and other information that shows whether the IP addresses to be checked are malicious. The last one is the reasoning module which is fuzzy logic-based. This module receives IP information from the previous module then analyses the information based on predefined rules to decide whether the checked IP is a potential source of malicious traffic or not.

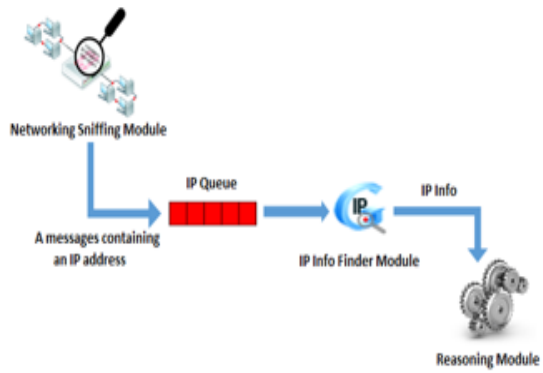

Figure 3. The identity checker block diagram

5.2 Network Sniffing Module

Network traffic is monitored using TcpDump tool. It has been decided to choose this tool as it is has distributions over many operating systems. In addition, it is a command line which simplifies the integration process with other modules. Moreover, it can be used with software such as wire shark to obtain a graphical representation. TCP dump reads network packets then parsed to extracts IP addresses, it then push messages in a queue that will be consumed by the next module (IP info finder).

5.3 IP Information Finder Module

This module gets information about IP participating in the traffic using web services (Neutrinoapi and fraud lab) [13, 14]. The information obtained includes IP geographic location, whether the IP in a block list, if the IP is an anonymous proxy, if the IP is an exit tor node, and the average IP rating which has a value between one and three (one is a the lowest rate and three is the highest). The geographic location will be checked against a predefined list of countries known with high volume of malicious traffic (the list will be referenced later as the malicious geographic list).

5.4 Reasoning Module

As mentioned earlier, the reasoning module is fuzzy logic based. The reasoning module receives its inputs from the previous module (IP info finder) and analyse them based on defined rules in order to decide whether the IP is malicious or not. The four elements of fuzzy logic have been implemented as follow:

1. **The Fuzzifier:** The membership function selected (is IP in the malicious geographic list, is IP an anonymous proxy, is IP a tor exit node, and is IP block listed) is a singleton function, as those inputs are Boolean values. The selected membership function for IP rating is specified using triangle functions, as shown in Fig 4.

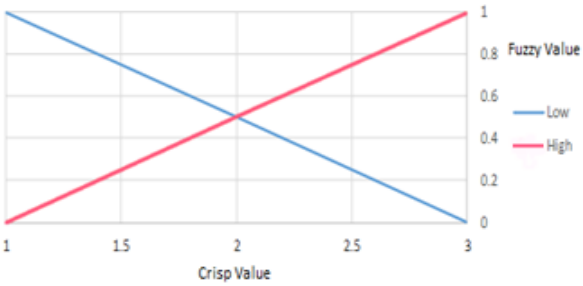

Figure 4: The selected membership function for IP rating

2. **Rule Base:** This is the part that contains the logic of producing the output. The rule base in this module contains four if-then rules as shown in Table 1.

Table 1. If-then rules used in the reasoning module

If condition	Then statement
(IP in a block list)	Possible malicious traffic
(IP country in the malicious geographic list) AND (IP is an anonymous proxy)	Possible malicious traffic
(IP country in the malicious geographic list) AND (IP is a TOR exit node)	Possible malicious traffic
(IP Rating is low)	Possible malicious traffic

The first rule is straightforward, the IP will be considered as a malicious one if the IP address is found in a block list. Finding an IP in a block list means that the IP address has been reported as having been used in malicious activities. The second and third rules check two parameters. One of them is whether an IP is on the malicious

geographic list or not. It is not practical to consider an IP as a malicious one if it is only located in one of the countries found in the malicious geographic list, as there may be legal traffic from these countries. Anonymous proxies and tor are used in a way that enables users to protect access to the web anonymously. Attackers normally do not need to be in the listed countries, they direct their traffic through a proxy or tor located in one of those countries. Therefore, getting traffic from anonymous proxies or tor-exit nodes located in those countries raises an alert of potential malicious traffic. The last rule checks the average IP rating. The IP address will be considered malicious if the average rating is low.

3. **Defuzzifier:** The selected membership function for the output is as shown in Fig 5. The output represents the probability of having malicious traffic from the checked IP address. If the probability is higher than 0.5, the IP will be considered as malicious. Otherwise, it will be considered as normal.
4. **Inference Engine:** The inference engine can be considered as the heart of reasoning, as it is responsible for mapping given inputs to a fuzzy output, using the specified rules. The inference engine used in this module is Mamdani, which is commonly used in fuzzy logic systems [15].

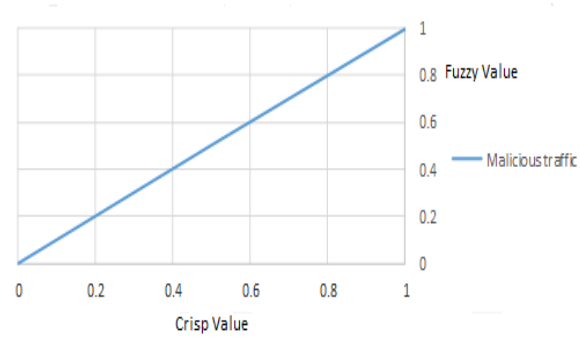

Figure 5: The membership function for the ouput

6. Identity Checker Evaluation

6.1 Evaluation Approach

The identity checker was evaluated using a metrics based approach [16]. The approach looks at intrusion detection systems from different angles, and it includes logistics, architectural, and performance metrics. The logistic metric evaluates the system in terms of maintainability, manageability, and dependency. The design metric is used to find how well the system performs in terms of resources consumption, integration, and speed. The last metric used in this approach is the confusion metric (performance metric), which finds how well the system does its job (detecting multi-stage attacks) in the form of true positive, true negative, false positive, and false negative. Each category in the logistic and design metrics will have a score between one and three (one is the lowest and three is the highest) based on number advantages and disadvantages For example, consider evaluating the system throughput. The system will score one If it has a low throughput while it will score two if it has a high throughput but with consuming a lot of hardware resources. On the other hand, the system will score three if it has a high throughput without consuming a lot of hardware resources.

6.2 Logistics Metrics

Table 2 shows the score for each item in the logistic assessment. The score for the distributed management item is two, as the system supports it but with some potential issues in the buffering area. The score for ease of configuration is two as many of its components can be easily installed but the configurations is not centralized in one user interface and is scattered over different areas. In addition, some components require prior knowledge to get installed. The score for ease of policy management is also two, as detection rules can be easily changed by using the same inputs. The score for outsource solution is poor (one), as the system is found to be massively dependent on using web services. The score for platform requirements is three, as the system supports running on different platforms, and its hardware requirements are dependent on network volume traffic.

Table 2. Logistics metrics

Item	Score
Distributed Management	2
Ease of configuration	2
Ease of policy management	2
Outsource Solutions	1
Platform Requirements	3

6.3. Design Metrics

Table 3 shows the score for each item in the design assessment. The score for adjustable sensitivity is two, as it supports adjusting sensitivity through modifying the fuzzy rules but is associated with some difficulties in some scenarios. The score of data storage is three, as it does not require less than one Megabyte to store fuzzy rules and blacklisted countries in a database. The score for multi-sensor support is three, as it has the ability to communicate with different sensors, other than the one proposed with the system. The score for both firewall interaction and incident logging/notification is also three, as the system is an open source PHP code that can be easily modified. The score for packet loss is two, as TCPDUMP cannot perform well in high-speed networks. The system

throughput on the testing environment has not achieved a high rate abut it is acceptable (around 10 packets/second), so the score will be two for this item.

Table 3. Design Metrics

Item	Score
Adjustable sensitivity	2
Data Storage	3
Multi sensor support	3
Firewall Interaction	3
Incident Logging and notification	3
Packet loss	2
System Throughput	2

6.4. Performance Metrics

The performance was first tested using a list of 91,744 IP addresses (10.99% Normal, 0.57% anonymous proxy in a black listed country, 88.53% block listed IP addresses) to ensure that the solution was capable of distinguishing between malicious and normal IP addresses. The results obtained were as shown in table 4.

Table 4. Confusion Metrics

Class	True Positive	False Negative
Normal	1.00	0.00
Malicious	0.9984	0.0016

The solution was then tested with four different multi-stage attack scenarios (SQL attack, Cross site scripting, Dictionary attack, and UDP scan) [11] using their trace files. The solution was able to predict three of them (SQL attack, Dictionary attack, and UDP scan) from the first packet, while it failed to detect the cross site-scripting scenario as none of the IP addresses participating in the traffic was categorised as malicious.

7. PQS Based Model

7.1 Overview

As discussed in the previous section, the identity checker fails to predict attacks if IPs involved in the traffic are not categorized as malicious. In this case, the traffic contents need to be analysed in order to capture any malicious activity. The following was reported [17] about conventional network security tools:

"Conventional network security applications fail to provide the security analyst with a central repository and correlated view of attack data as observed by different systems, processes, etc. In situations where the analyst oversees an enterprise-sized network, critical information can be easily overlooked due to massive overload of data from deployed security applications".

Process Query Systems (PQS) is highy capable in terms of fusion and correlation of malicious network activities in addition, it can track multiple attacks simultaneously. Therefore, it can be employed to detect the sequence of security state transitions occurring during multi-stage attacks [10]. One of the systems based on PQS is PQSNet that uses sensors established around network such as IDS, web logs, and firewalls. The data provided by sensors are parsed into observation then provided to PQS that updates the states of processes.

7.2 Sensors

As mentioned in the previous section, PQS are fed by sensors. In this module, a number of sensors will be used based on attacks that will be modelled. Therefore, the following sensors will be used:

- Network sniffer: It can be used as a sniffing module that tells about network traffic activities. For example, it can provide the protocol and port used in communication. It can be also used to check the contents of received packets in some cases. TCPDUMP or SNORT can be used as a network sniffer
- Web log: Web server logs from APACHE can tell whether there are some web requests containing some suspicious values. It can also provide all errors returned by the webservers
- DB log: it can provide information about number of some queries on a specific table (e.g. user credential table) within a specific period of time.

7.3 Processes

In this context, attacks are considered processes. Four attack scenarios will be considered in this paper for modelling as examples of using a PQS approach in detecting multi-stage attack scenarios. Any other attack model can be then added to PQS without the need to change other models. That gives an advantage of PQS over a rules-based approach that requires updating many rules when considering new attacks.

The first scenario is the scenario that was discussed earlier in this paper. This scenario passes through multiple states. The first one is triggered by sending a DNS query. The second one is triggered by receiving an irregular DNS response. The third state

is attempting to communicate (TCP handshake) with one of the IPs in the irregular DNS response (it may be more than one attempt). Once the communication is established, bot net commands are sent. The PQS model of this attack scenario consists of three states (A, B, C) and three observables which are DNS query (a), irregular DNS response (b), and TCP handshake with one of the IPs returned in the DNS response (c) as shown in Figure 6 (Model 1). An alert will be raised if state C is reached.

The second scenario that will be considered for PQS modelling is scenario C (Carriage Return Line Feed injection) that was discussed earlier in this paper. The first state is associated with a scan for PHP configurations. The second state is triggered by receiving a request containing special characters (CRLF) that can be used for splitting headers. The last state is triggered by requesting a file from an external server. The last state and event will be excluded from the modelling as the system should not reach that state, which indicates the occurrence of the attack. Therefore, modelling this scenario will involve only two states (D, E) and two observables which are a PHP configuration scan (d) and receiving a suspicious web request containing CRLF characters (e) as shown in Figure 6 (Model 2). An alert should be raised once a suspicious web request is received even if it is not preceded by a scan.

The third scenario that will be considered in PQS modelling is the scenario that the identity checker model failed to detect (the cross site scripting scenario). This scenario is very similar to the previous one. It can be detected once a web request is received that contains some tags (HTML or JavaScript tags). This step may be preceded by a web request that calls the page experiencing a cross-site scripting vulnerability. Therefore, this scenario can be modelled involving two states (F, G) and two observables; normal web request (f) and suspicious web request containing HTML or JavaScript tags (g) as shown in Figure 6 (Model 3). Similar to the previous model, an alert should be raised once a suspicious web request is received even if it is not preceded by a normal web request.

The last scenario is an SQL attack. In this scenario an attacker tries to send a huge volume of web requests from different machines in a very short period of time. These web requests target web pages that contact a database. That creates a huge load on the database server causing it to go down. Modelling this scenario will include two states while the observables will be defined as an SQL query requested from the web server. The transition from the first state to the second state, which indicates the occurrence of an SQL attack, will be triggered by a number (will refer to it as n) of an SQL query requested from the web server occurring within a very short period of time as shown in Figure 6 (Model 4).

Table 5 shows all observables involved in the processes and their sources. In other words, the table shows from which sensor each observable is monitored.

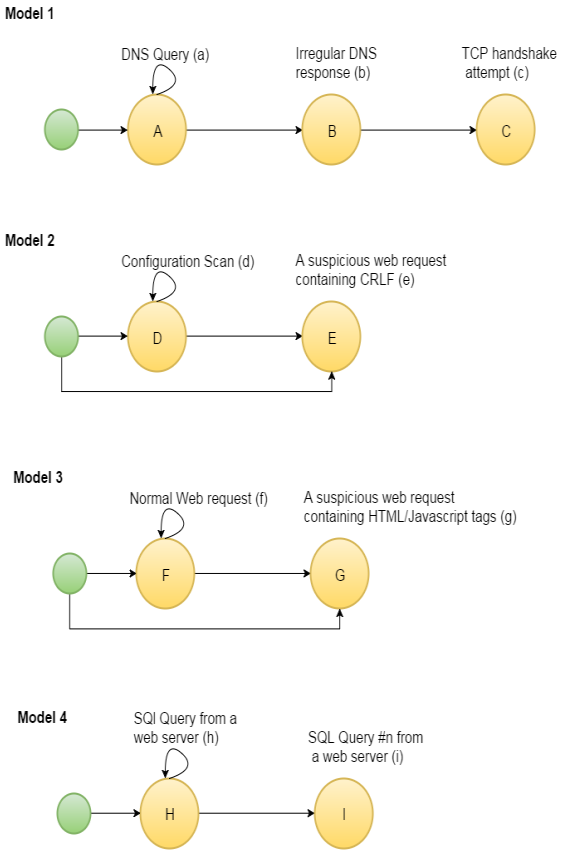

Figure 6: The membership function for the output

Table 5. List of observables

Observable	Description	Source
a	A DNS query	Traffic sniffer
b	Irregular DNS query response	Traffic sniffer
c	TCP handshake attempt	Traffic sniffer
d	Configuration scan	Web logs
e	A suspicious web request containing CRLF	Web logs
f	Normal web request	Web log/Traffic sniffer
g	A suspicious web request containing HTML/JavaScript tags	Web logs
h,i	SQL query from a web server	DB logs

8. Related Work

A number of research studies have been conducted in the multi-stage attacks detection area. One of the studies [18] proposes a correlation framework that combines two engines, online and offline, and uses two mechanisms, high quality knowledge-based and statistical-based correlation. The proposed framework achieved a 92% multi-stage detection rate and 21.8% false positive rate during their lab experiments. This approach reduces the computation expenses by analysing only alerts received by IDS. However, the massive dependence on alerts received by IDS may lead to missing capturing attacks if alerts are not received.

Another study [19] proposed a system that follows the attack scenario construction approach. This approach is based on associating two security incidents, and it tries to find consequences of one incident and prerequisites for the incident that may occur later. The strong point of this approach is the ability to construct new attacks created by a mixture of known attacks that can be detected. On the other hand, attacks cannot be tracked without finding cause and effect of these attacks. Moreover, it requires a large consumption of computer resources.

Another study was based on using Hidden Markov Models (HMM) [20]. This study found that the HMM approach achieved greater classification accuracy, compared to other approaches. However, they reported that the accuracy obtained was at the expense of additional computations.

The proposed solution has an advantage over the above-mentioned solutions by checking the identity and the traffic contents rather than the traffic contents only. However, it may require more hardware resources as it has got two components. In addition, the complexity of the system is highly dependent in optimizing the number of models added to the system.

9. Conclusion and Future Work

The proposed approach in this paper to detecting multi-stage attacks is based on a hybrid approach that involves evaluating IP addresses participating in monitored network traffic using fuzzy logic. In addition, it involves using the PQS approach, which checks the traffic contents. The identity checker (IP info-based component) has been evaluated individually using a metrics-based approach. It has a medium score from the logistics perspective. On the other hand, it has a high score when looking from the design perspective. The last part of the evaluation looks at the system performance, and it was found that the system achieved a good performance with zero false positive and a high detection rate. However, it fails to detect multi-stage attacks if IP addresses participating in the traffic are not classified as malicious IP addresses. Such cases will be handled using the PQS approach.

It is planned to add more models to the PQS-based components then evaluating individually using the metrics-based approach. When adding more models, optimizing the number of models by combining similar models within one model will be considered in order to improve the performance of the system overall.

10. References

[1] D. Clark, “The Problem isn't Attribution; It's Multi-Stage Attacks” the Re-Architecting the Internet Workshop, 2010, Article No.11.

[2] J. Muila, A Novel Intrusion Detection System (IDS) Architecture, 2010.

[3] Tal Global. (2011) Operation Shady Rat – What It Really Means, and What You Can Learn From It? [Online] Available from: http://talglobal.com/operation-shady-rat-what-it-really-means-and-what-you-can-learn-from-it/. [Accessed: 19th Feb 2015]

[4] S. Rajasekaran, G.A. Pai, “Neural Networks, Fuzzy Logic and Genetic Algorithm: Synthesis and Applications” 2003, PHI Learning Pvt. Ltd.

[5]P. Alberto, A. Sala, and M. Olivares, “Fuzzy Logic Controllers. Methodology. Advantages and Drawbacks” [Online] Available from: http://www.softcomputing.es/estylf08/es/2000-X%20Congreso/01%20SESION%20INAUGURAL.pdf [Accessed: 19th April 2015]

[6]K. Pulo, “Fuzzy Logic vs Machine Learning” , [Online] Available from: http://www.kev.pulo.com.au/ai/ fuzzyml_report/ [Accessed: 20th April 2015]

[7]A. Chitrey, “prehensive Study of Social Engineering Based Attacks in India to Develop a Conceptual Model” IJINS ,2012. vol.1, n.4

[8]B. Hall, 2011 Countering Web Injection Attacks: A Proof of Concept, MSc thesis, University of Manchester UK.

[9]G. Cybenko, et al., "An overview of process query systems", Proc. SPIE 5403, Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense III, 183 (September 15, 2004)

[10]G. Cybenko, and V. Berk (2007): “Process Query Systems”, IEEE Computer

[11]TP Group, Network Trace Files [Online] Available from: http://www.tp.org/jay/nwanalysis/traces/General%20Trace%20Files/ [Accessed: 24th Feb 2015]

[12]Mcafee. Operation Shady Rat – What It Really Means, and What You Can Learn From It? [Online] Available from:http://www.symantec.com/connect/blogs/truth-behind-shady-rat/ . [Accessed: 25th Feb 2015]

[13]Neutrino API [Online] Available from: https://www.neutrinoapi.com/. [Accessed: 25th Feb 2015]

[14]Fraudd Lab API [Online] Available from: https://www.fraudlabs.com . [Accessed: 25th Feb 2015]

[15]Mathworks Operation Shady Rat – What Is Mamdani-Type Fuzzy Inference? [Online] Available from: http://uk.mathworks.com/help/fuzzy/what-is-mamdani-type-fuzzy-inference.html. [Accessed: 26th April 2015]

[16]G. A. Fink, "A Metrics-Based Approach to Intrusion Detection System Evaluation for Distributed Real-Time Systems" Information TransferTechnology Group, 2002, Code B35, Naval Surface Warfare Center, Dahlgren Division

[17]I. Greogrio, et al.., "Detection of Complex Cyber Attacks" Proc. SPIE 6201, Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense V, 620106 (May 10,2006); doi:10.1117/12.670131

[18]F. Alserhani, M. Akhlaq, I. Awan, A. Cullen, and A. Mellor (2009): "Multi-Tier Evaluation of Network Intrusion Detection Systems" Journal for Information Assurance and Security (JIAS), 5 (4): 301-310.

[19]S. Templeton and K. Levit. A requires/provides model for computer attacks. In Proc. of New Security Paradigms Workshop, pages 31 – 38. September 2000.

[20] D. Ourston, S. Matzner,W. Stump, and B. Hopkins. Applications of hidden markov models to detecting multistage network attacks. In Proceedings of the 36th Hawaii International Conference on Systems Sciences, Los Alamitos, CA, USA, 2003 2003. IEEE Comput. Soc. 36th Hawaii International Conference on Systems Sciences, 6-9 January 2003, Big Island, HI, USA.

