2018

Construction of a custom network security appliance

Jacob Rickerd

Construction of a custom network security appliance

Abstract

Over the last three semesters, I worked toward my final goal to develop a custom network security appliance. I first began by completing a comparison analysis of network intrusion detection systems which are devices that read traffic from the network and determine if network packets should go through or be dropped. Second, I conducted a feasibility study of a custom framework to profile attackers in a network; this yielded positive results. Finally, I worked on creating a custom network security appliance; it uses the profiles I created in my framework to more efficiently block malicious attackers in comparison to other security appliances. Below is a more detailed account of how these steps were taken to result in my final security appliance.

Degree Type

Open Access Senior Honors Thesis

Department or School

Technology and Professional Services Management

First Advisor

Dr. James Banfield

Second Advisor

Dr. Suleiman Ashur

Keywords

Network Intrusion Detection System, Security, Custom Security Appliance, Machine Learning Security

Subject Categories

Information Security

CONSTRUCTION OF A CUSTOM NETWORK SECURITY APPLIANCE

By

Jacob Rickerd

A Senior Project Submitted to the

Eastern Michigan University

Honors College

in Partial Fulfillment of the Requirements for Graduation

with Honors in Information Assurance

Approved at Ypsilanti, Michigan, on this date 7 May 2018

Supervising Instructor, Dr. James Banfield

Honors Advisor, Dr. James Banfield

Department Head, Dr. Suleiman Ashur

Honors Director (Print Name and have signed)

Table of Contents

Abstract

Over the last three semesters, I worked toward my final goal to develop a custom network security appliance. I first began by completing a comparison analysis of network intrusion detection systems which are devices that read traffic from the network and determine if network packets should go through or be dropped. Second, I conducted a feasibility study of a custom framework to profile attackers in a network; this yielded positive results. Finally, I worked on creating a custom network security appliance; it uses the profiles I created in my framework to more efficiently block malicious attackers in comparison to other security appliances. Below is a more detailed account of how these steps were taken to result in my final security appliance.

Keywords

Network Intrusion Detection System, Security, Custom Security Appliance

Introduction

Computer networks today get flooded with an immense amount of attacks. These systems are exposed to so many different types of threats, having damages ranging from minor losses to the destruction of the entire information system (Jouini, 2014). Various types of threats that can cause different types of damages on information systems, leading to large financial losses (Bhuyan, 2014; Jouini, 2014, 2015). Getting visibility into a computer network can be difficult for many network administrators; understanding the threats to their information systems and how to combat the attacks proves to be even more difficult than having visibility into the network (Jouini, 2014). Information security

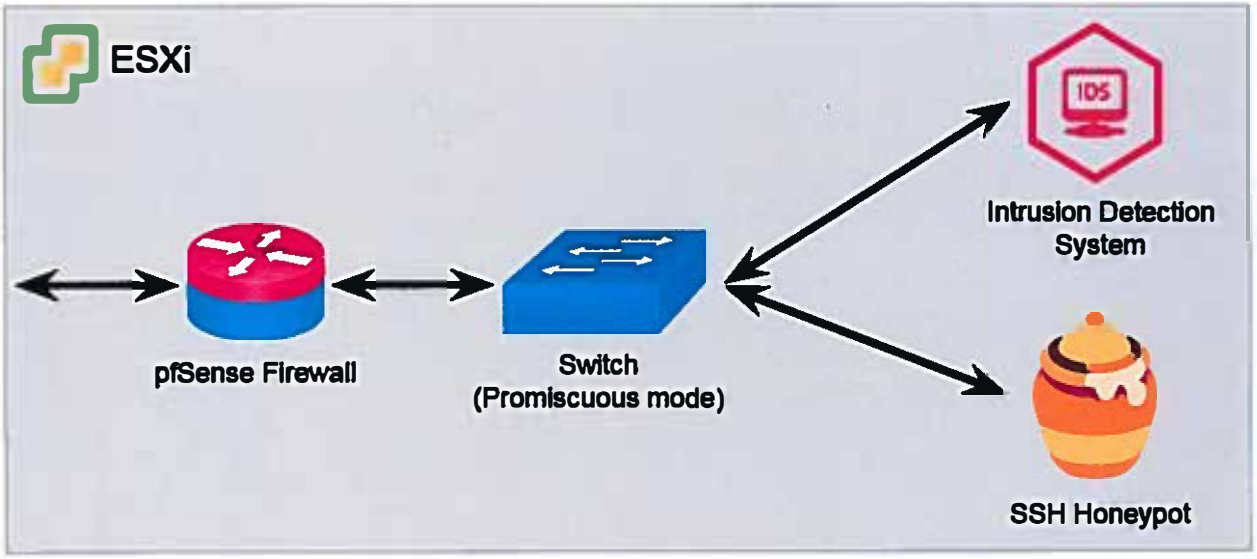
is the most difficult part of processing information (Jouini, 2015), therefore, it is important to study these concepts and learn how to advance them. There are already tools to monitor networks including dozens of options, some paid and some free (Pawar, 2015). However, none of these options accomplish everything that is needed in order to create an accurate and verbose report of a network's activity.

Due to the importance of understanding how to gain visibility into a network, understanding the threats on a network, and learning how combat those attacks, I have devoted my upperclassman years of school to learning the complexities of these problems. This past year and a half, I have begun my study of these topics, spending time working on three phases of research: a comparison of network intrusion detection systems, the feasibility of a custom framework that profiles attackers on a network, and creating a custom security appliance using the framework for network administrators to use to gain insight into the network's activity.

Phase One: A Comparison Analysis of Network Intrusion Detection Systems

Phase one, in more detail, focused on network intrusion detection systems; I studied three of them, all with different sets of features, to determine which of them was most successful in detecting malicious traffic on the network. Determining which of these three types of devices allowed the most visibility into the network is important to the remaining portions of my research as it is imperative to understand how threats can be represented and what types of threats may occur (Jouini, 2014). The intention with these devices is that users will create their own rules to detect attackers, but my study was to

determine which of these was most helpful with their default rules. This is important to understand because it gives a basis for how network traffic analysis works both technically and conceptually. By doing this portion of my study, I was able to come up with my own plan for creating a security appliance which lead to phase two of my research project.



The above diagram illustrates the layout of the network used during this phase of study.

The honeypot used in this study is called Cowrie and is run on a Debian based Linux operating system. The Linux machine was given a public IP address and was hosted within a VMware ESXi instance running under the Information Assurance department. VMware ESXi is an enterprise-grade virtualization software that allowed me to create several servers, all within a single physical machine. The system ran and collected data for one month, after which analysis occurred. Because I captured human network traffic during this study, I submitted an application to the Human Subjects

Review Committee and it was approved. The data collected in this study was used in my future research.

The honeypot deployed, mimics an SSH service using the Cowrie honeypot. SSH, or Secure Shell, is a network protocol that allows clients to connect to servers in order to run commands remotely. Cowrie, the software used to create the honeypot, was setup with the username "root" and password "123456". This is a very easy password and allows for a large number of attackers to breach the system. Once they are in, the Cowrie honeypot stores any malware that the attacker attempted to download.

In addition to the traffic reaching the honeypot, it will also reach the network intrusion detection system that is on the network. This analyzed all the traffic and tried to detect attacks or malware. The network intrusion detection system kept a log of everything that caused an alert as well as saving network data for further review. This captured network traffic was then replayed through the remaining network intrusion detection systems in order to log their alerts. This ensured the traffic is the exact same through each of the network intrusion detection systems.

Phase Two: Framework for Classifying Attack Profiles

Phase two, is focused on the feasibility of a custom framework that profiles attackers on a network. I started by first correlating the actions taken in the network back to each attacker. This allowed me to see exactly what each attack did in the network over the thirty day period. This was done through a combination of manual analysis using Wireshark, a popular network traffic analyzation tool, and automated tools and scripts. Some of these tools included: intrusion detection systems; network security monitors

such as Bro, devices that give a high-level overview of what is happening in the network, which further break down events in the network; and custom programs that I wrote to correlate events, that took place on the service they hacked, back to the original hacker.

Key indicators that I used to profile these attacks included data such as the version and type of tool used, country of origin, commands executed, number and type of passwords attempted, type and purpose of malware (if any) downloaded on the system, length of time between connections, and any other factors discovered after initial analysis. After studying these indicators, I was able to tell, with strong certainty, if the connection is coming from a real human or if it is an automated bot. This was a very important distinction because bots often operate the same no matter the network on which they are attacking. If I could successfully develop a framework to profile these bot attacks, then detecting them in other networks should be easy as well. My profile proved to be feasible and was reviewed and prepared for the final phase of research.

Phase Three: Construction of a Custom Network Security Appliance

Due to phase two of my project proving feasible, I used the framework I created to conduct phase three, the final phase and my senior project, which was to create my own network security appliance based on the findings of my previous research projects. I incorporated the framework I created in phase two into my network security appliance. In addition to the framework, the custom network security appliance I am creating incorporates many industry standard anti-malware and anti-hacker techniques. Some of these concepts can be referenced in Jouini's study which proposes a model of a security appliance that focuses on making a systematic, extendable, and modular system (2015).

This project is the culmination of all my previous research. As an Information Assurance major and aspiring Security Engineer, knowing the complexities of network security appliances are imperative to my career, especially considering I learned how to make my own, a skill that is very unique.

To create this custom security appliance, I decided to wrap my framework around an already popular network intrusion detection system. This would allow more people to use the appliance if the barrier to entry was lower. The network intrusion detection system I decided to go with is called Suricata, and was the best open source network intrusion detection system per the results of phase one of my research.

The next step to creating this appliance was to be able to automatically create profiles based on data outputted by the Cowrie honeypot. To do this, I once again turned to Python. The program I wrote will read in the log files from the Cowrie honeypot and create profiles based on my framework.

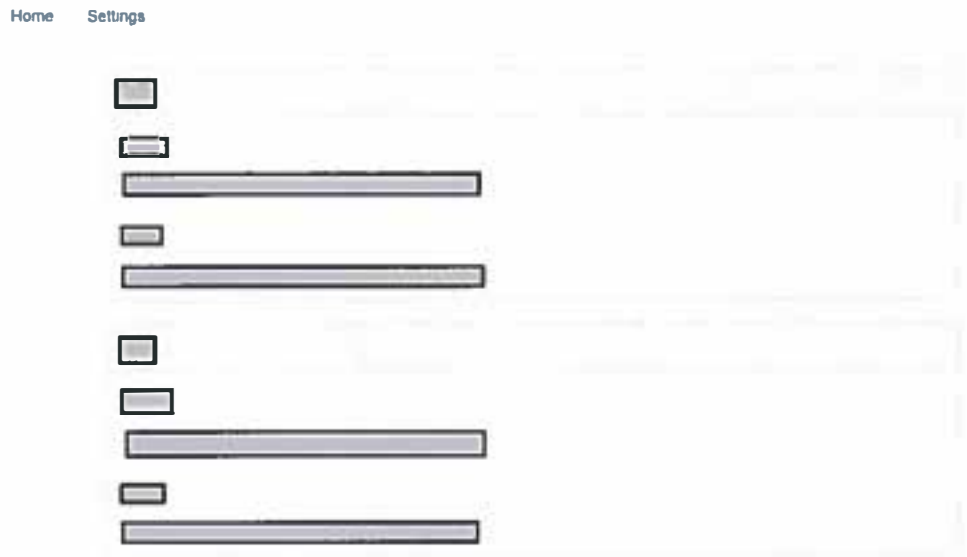
To create these profiles, I decided to focus on one specific area of the data I collected in phase one: download information. As part of the data I collected in phase one, I have information about what each attacker downloaded including: file hashes, file names, download origins, and file contents. Using this information, I was able to see if any attackers downloaded a file with matching characteristics of another. For example, I would be able to tell if an attacker downloaded the same file but from a different server as another attacker, downloaded a different file but from the same server, downloaded a file with the same file name from any location, and more.

Using this information, I was able to automatically correlate several attackers into a single profile. One of my most extreme examples of this framework in action resulted

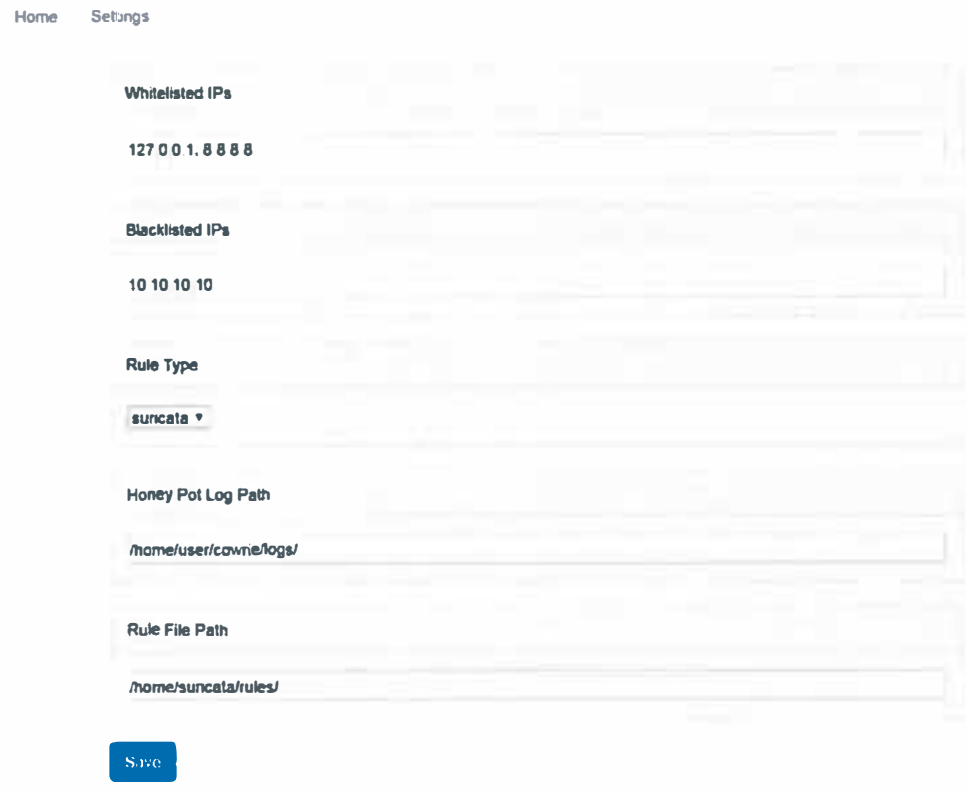
in a profile with 34 different attackers in it. These attackers had all downloaded a file called "bins.sh" from various locations. Out of these 34, some of these files were identical and others were very similar in nature, allowing me to conclude that they would accomplish a similar task.

After generating these profiles, the next step is to output this information in a way the network intrusion detection system (in this case Suricata) could use. To do this, I take the IP addresses from the profiles and output a rule file that the network intrusion detection system can ingest that tells it to trigger if it sees traffic from any of those sources.

Once the framework to successfully create profiles was in place, I developed a web application frontend for the appliance using ReactJS (a client side JavaScript framework created by Facebook). To bridge the gap between the frontend and backend services, I used Flask, a popular Python RESTful API service. This allows the frontend ReactJS code to make calls to the backend, get the data, and perform actions. This frontend allows users to easily view the profiles generated by the framework and configure the appliance with ease. Some of the features of this frontend include a settings page allowing IP addresses to be whitelisted or blacklisted and a main overview page meant to display the current profiles generated from the framework. To ease in initial configuration, the settings page also allows the user to select the directory in which the appliance will read the log files from the honeypot. Additionally, the user can also specify the rule type the appliance should output in (currently it only supports Suricata), and the directory to which the rules should be outputted.

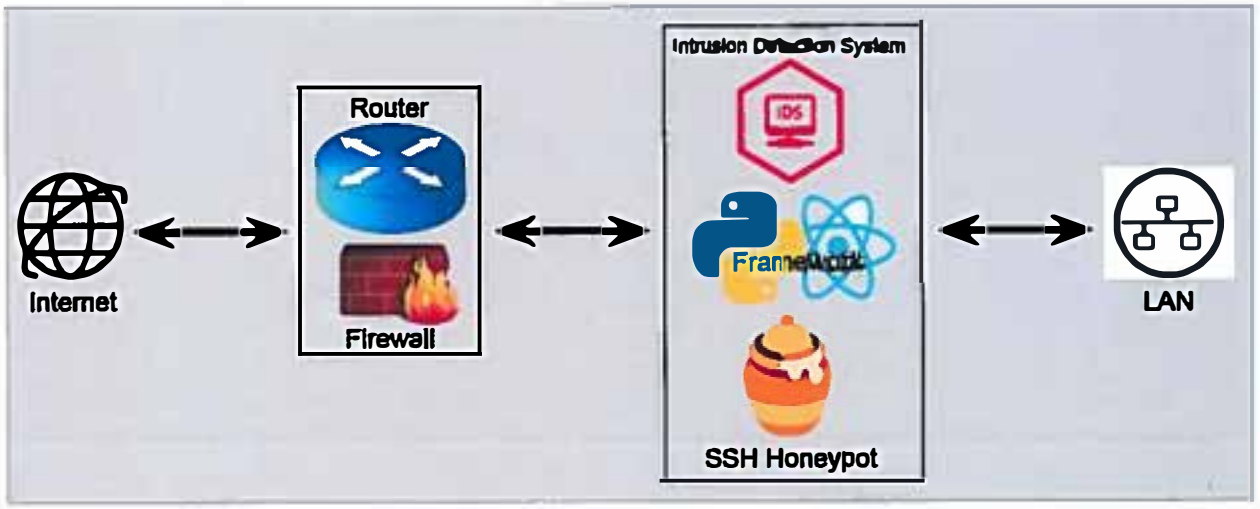


The above screenshot shows what the home page looks like when a user accesses the web application. Unfortunately, I have had to redact the data for security and privacy purposes.



The above screenshot shows what the settings page looks like when a user accesses the web application. They are able to configure all of the fields shown and their configurations will be saved.

To finish the overall configuration of the appliance, Suricata will have to be installed in the system and configured in inline mode. Then, the user will setup a Cowrie honeypot on the system and configure it to output its logs in JSON format. Next, the user will install my framework and configure it using the frontend web application. Finally, the user must setup a cron job to specify how often the framework should run to generate new rules. The recommended time is once every 24 hours.



The above diagram is an example of what the final setup for this custom network security appliance may look like. The appliance will sit behind the router/firewall in a network and have all traffic pass through it before it reaches the LAN. The honeypot and framework are installed and running on the same box as the network intrusion detection system. These three pieces together form the custom network security appliance.

Future Research

There are a lot of next steps that can be taken with this research. As of right now, the appliance will only automatically generate profiles based on downloaded data. However, I have many more data points about hackers that can be used. Future research would include the automatic creation of profiles based on this additional data as well as the downloaded data.

In addition to supporting Suricata, I would like this appliance to be able to support a wider range of popular open source network intrusion detection systems. The other most popular one is Snort. With support for Snort, I believe that even more people would be able to use and benefit from this framework.

Another idea for future research on this topic could include taking the logic used to make the profiles out of the Python framework and incorporate it directly into rules in the network intrusion detection system. For example, I mentioned earlier that I can correlate malicious attackers by attributes related to what they downloaded. Instead of tying all that information into a profile and outputting the IP addresses to block in the network intrusion detection system, one could create rules that allow the network intrusion detection system to make that correlation itself. Most network intrusion detection systems allow for more advanced rule types so this should be possible with some more engineering.

Link to Open Sourced Project

https://github.com/Rickerd0613/NetworkSecuritvAppliance

References

Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: methods, systems and tools. IEEE communications surveys & tutorials, 16(1), 303-336.

Jouini, M., Rabai, L. B. A., & Aissa, A. B. (2014). Classification of security threats in information systems. *Procedia Computer Science*, *32*, 489-496.

Jouini, M., Rabai, L. B. A., & Khedri, R. (2015). A multidimensional approach towards a quantitative assessment of security threats. *Procedia Computer Science*, *52*, 507-514.

Karim, A., Salleh, R. B., Shiraz, M., Shah, S. A. A., Awan, I., & Anuar, N. B. (2014). Botnet detection techniques: review, future trends, and issues. Journal of Zhejiang University SCIENCE C, 15(11), 943-983.

Pawar, M. V., & Anuradha, J. (2015). Network security and types of attacks in network. Procedia Computer Science, 48, 503-506.