Automation with Python Security Scanning Project

01/28/2022

MITRE
Watt Sengkhyavong
wsengkhyavong@mitre.org

Arvind Baskar
abaskar@ucdavis.edu, Masters Student, University of California, Davis
Pon Izhanathi Sundara Arumuganathan
natsu@ucdavis.edu, Masters Student, University of California, Davis
Tarang Gujar
tsgujar@ucdavis.edu, Masters Student, University of California, Davis
Kaustubh Chakravarthula
kchakravarthula@ucdavis.edu, Masters Student, University of California, Davis
Karan Mehta
khmehta@ucdavis.edu, Masters Student, University of California, Davis

Matt Bishop
mabishop@ucdavis.edu, Supervising Faculty, University of California, Davis

Domain: Security automation for Vulnerability detection and mitigation in Network devices

Keywords: Security Automation, Intrusion Detection Systems, Routers, Switches, Servers, Networks

The time to mitigate a security violation is increasing day by day due to new kinds of attacks and manual triage taking a lot of time. With automation, this time can be greatly reduced and the remedial measures can be easily implemented with the help of automated measures taken upon recognizing an incident.

Team Responsibilities:

Pon Izhanathi Sundara Arumuganathan: Report Writing, Lab Environment Creation, Mitigation Automation

Arvind Baskar: Report Writing, Lab Environment Creation, SPLUNK log capture, Mitigation Automation

Karan Mehta: Report Writing, Ansible setup for running the python scripts, SPLUNK log capture

Kaustubh Chakravarthula: Report Writing, Ansible setup for running the python scripts, Alerting system

Tarang Gujar: Report Writing, Vulnerability and IDS Analysis along with implementation, Alerting system

Total Budget: $47,782.8. Please see the details in the “Budget” section below.

Project Deliverables:

- Final Project Report: Understanding, detecting and mitigating vulnerabilities in a network using security automation. Expected Delivery Date: 04/29/2022.
- Project Presentation and Demo: The team will give a presentation and a demo of the project to all the other teams and technical directors. Expected Delivery Date: 04/27/2022 - 04/28/2022.
- Weekly Updates: Updating the Technical director with weekly progress reports.

Executive Summary

Automation with Python Security Scanning Project Proposal, January 28, 2022

Vulnerabilities are fundamental flaws in the system through which an attacker can gain access and use them to either exploit, cause damage or manipulate the system somehow. Cybercriminals are always waiting for an opportunity to exploit even the smallest of vulnerabilities. Their motivation to do this may range from breaking into systems for learning purposes to obtaining personal data like credit card information, social security number, etc. Therefore we need to always stay alert and be aware of all the possible vulnerabilities in our system. Vulnerabilities are broadly classified into three types:

- Network vulnerabilities - physical components(hardware) of a network are exploited and broken into
- Operating system(OS) vulnerabilities - breaking and gaining access to the assets built on top of the OS by identifying OS vulnerabilities
- Human vulnerabilities - taking advantage of human errors to obtain personal data for malicious purposes.

The most common vulnerabilities include:

- Weak passwords.
- Outdated software applications.
- Poor firewall configurations.
- Older versions of Operating Systems without security updates.

If not noticed and patched at earlier stages, it's challenging to recover from the system attacks caused by a few of these vulnerabilities. Hence it is a constant fight against time for all the security analysts to continuously scan and detect the entire system and release patches for its security before an attacker can find them. It is not feasible to regularly monitor these systems manually. Hence we aim to develop, test, and deploy an automated tool using Python programming language to periodically schedule the infrastructure scan for threats and vulnerabilities and send alerts as appropriate. We send an alert using a ticketing system like JIRA to the respective users. We run the scripts in batches periodically using Ansible to check for vulnerabilities constantly.

Keywords: Security Scanning, Security Automation, Intrusion Detection Systems, Routers, Switches, Servers, Networks

This project aims to develop an automation script using Python, which will continuously scan for vulnerabilities in the network not limited to Firewall, Routers, Switches, Windows and/or Linux Servers. We start by identifying the security use case and predefined vulnerabilities that can and should be automated. After which, we will introduce those vulnerabilities into the test environment and then develop a code to detect these vulnerabilities automatically. For every scan, we retrieve the hostname, IP address of devices/VMs, applications installed in them, OS information, and using all this data along with detected vulnerabilities.

Motivation

Security automation is machine-based execution of security actions, which can detect, investigate and remediate cyberthreats with or without human intervention. According to recent research, it takes an average of 46 days for the security team to respond to an attack. This involves:

1. Manually investigating the incident
2. Analyzing the data
3. Jumping between unintegrated systems during triage
4. Coordinating the response.

Many of these can be automated for quick response. Security analysts can then look into the more mundane but essential tasks. Hence, the need for automation has grown as the scope of attacks on a system has increased. We seek to create an automated system that can detect these attacks, visualize the impact, provide an alert, and suggest the next step to mitigate the attack and bring down the vulnerability as a patch to the system.

Previous Work

Security Automation and its evolution[2]: This article[2] explains the evolution in the definition of security automation. Barak explains four advanced elements considered in security automation other than the usual prevention and detection technologies.

1. Policy Execution - Nowadays, networks are growing in size thereby increasing their complexity. Therefore, manual management of security policies would be tedious and nearly impossible. So we could automate this policy. The author of this paper says that we can automate the administrative work done by the IT team, and this policy automation can help the IT teams gain greater control over data, costs, and time.
2. Alert Monitoring and Prioritization - Automation is usually viewed as monitoring and prioritizing results. Alerts need to be manually looked at and analyzed to determine thresholds and a set of rules to categorize a particular alert as important or not. Security analysts would need to stare at their screens all day for this analysis. But if this process was automated using behavioral analytics and threat intelligence, we could efficiently set the rules and thresholds with minimum human effort. This would also help keep the thresholds and rules up to date as cyber security threats are constantly changing, and the cybercriminals know the common alerts being looked at by the companies.
3. Incident Response Planning - The response workflow for companies in the event of a threat involves a long and tedious set of steps. Manually performing this response workflow would be a cumbersome task. Hence, we can automate this incident response task. We can automate a response workflow through a smart ticketing system when a threat is detected. Through this, they can also track the evolution of a security incident.
4. Investigation, Action, and Remediation - Security analysts perform investigation, action, and remediation in the process of threat detection and mitigation. This wastes lots of their time and cost. Therefore, we can automate this end-to-end process.

In our project, we will be automating all the above-mentioned mechanisms and will also add a ticketing system to the mitigation automation process.

Security Automation and Threat Information-sharing Options[3]: This paper explains the importance of information sharing in a secured manner and how the security community is working towards standardizing the security of information sharing. In this paper, Panos Kampanakis summarizes each existing security model as follows:-

1. TAXII and STIX: Trusted Automated Exchange of Indicator Information(TAXII) is coordinated by MITRE. The threat information exchanged varies from IP addresses, x-mailer fields and malware to discovered vulnerabilities and defensive courses of action. Structured Threat Information Expression (STIX) is an expressive, flexible and extensible XML-based language that conveys potential cyber threat information. STIX also uses other structured languages like Snort and Yara
2. SCAP: The Security Content Automation Protocol (SCAP) was developed by MITRE to define specifications that would be able to support information sharing between private and government organizations. It has an Asset Identification which is an XML schema and it provides a way to uniquely identify the known identifiers and assets.
3. CWE, CWSS and CEE: Common Weakness Enumeration (CWE) is a dictionary of unique identifiers of common software weaknesses. Common Weakness Scoring System (CWSS) scores software weaknesses using a specific metric. Common Event Expression (CEE) was developed to facilitate the exchange of electronic events and logs. Similarly, the author describes other models for data sharing like MARF, NEA, INCH, MILE, SACM, CVRF, OpenIOC, IF-MAP, VERIS. Although each model is improving over time with added extensions, the author believes that TAXII, STIX, OVAL, SWID, CVE, and CVSS are the models that will play a crucial role in data sharing. Depending on the use case, the ideal model may vary and should be chosen appropriately.

All the models described above use XML as their mode or format of representation. Along with its advantages, the XML also has some disadvantages and might fail in the case of information bloat, redundancy and processing. Since security automation aims at simplifying and speeding up the exchange, the author suggests using JSON or YAML instead as alternatives.

Since the security automation ecosystem is broad, there is bound to be an overlap between various data models. Based on the use-case(goals for data sharing) and functionalities of each model, we need to choose the appropriate data-sharing model. Considering all the above points mentioned by the author in his research, we will incorporate the best-suited data model for threat information sharing in the network.

Specific Aims

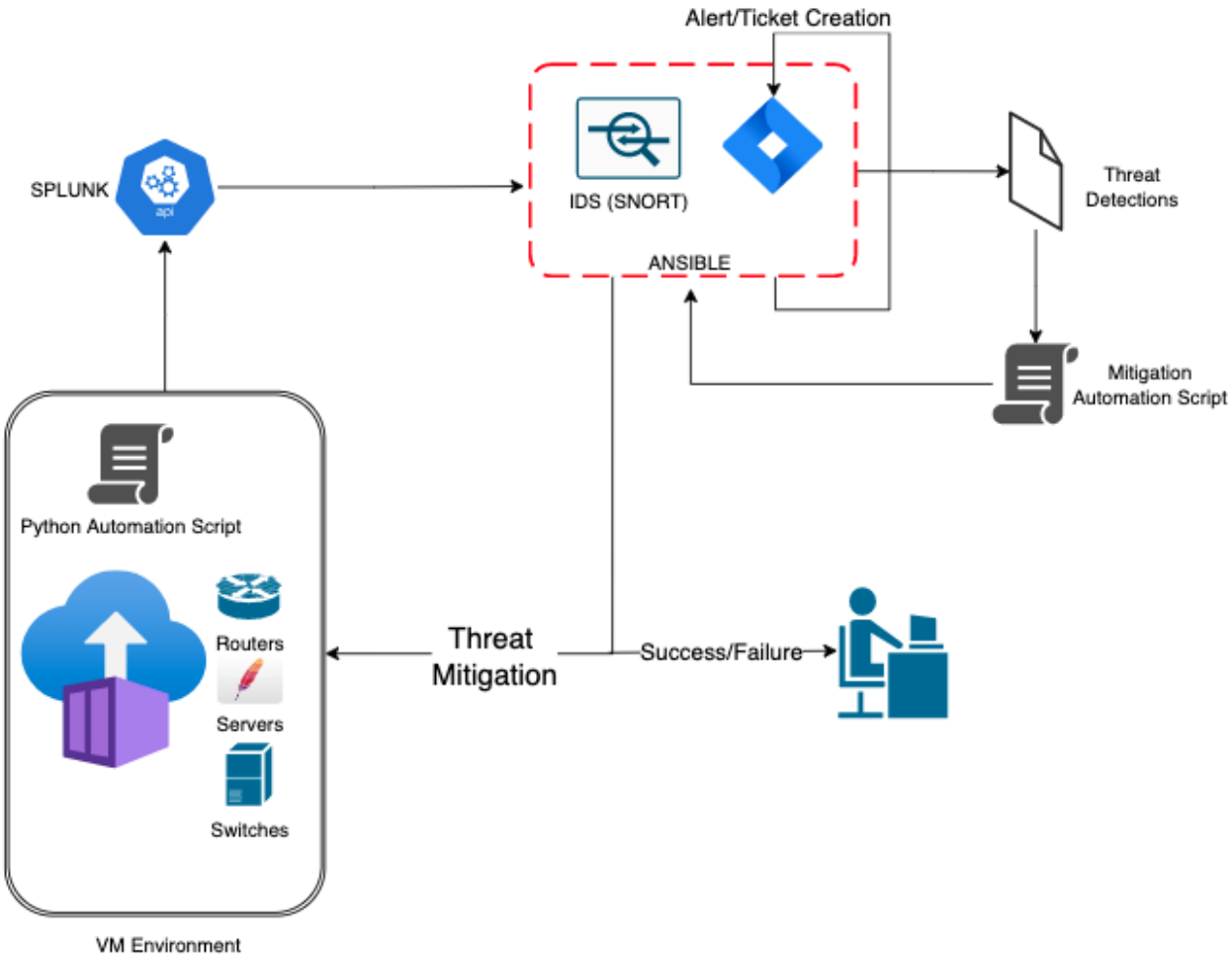
The goals of the project are:

1. Environment Creation - Create a virtual environment with some connected network devices.
2. Periodic Log Generation - Collect data and generate logs for the devices in the virtual network.
3. IDS and vulnerability detection - Identify vulnerabilities in the network devices.
4. Response Mechanism - Create tickets for the list of vulnerabilities for each device present in the virtual network.
5. Mitigation Automation - Automate threat mitigation workflow in case of an identified vulnerability.

Plan

Assumptions:

1. We assume that we have access to the network logs for simplicity purposes.



To achieve our specific aims successfully, we have created an architecture diagram as given above and the detailed description of the steps are given below:-

1. Environment Creation - Create a virtual environment on a cloud platform, consisting of various connected network devices such as routers, switches, and servers. Create vulnerabilities intentionally in the virtual devices to identify vulnerabilities.
2. Periodic Log Generation - Develop python scripts to collect and generate logs for every device in the virtual network. These scripts will then be run on AWX Ansible as batches periodically. We will be hosting it over a Kubernetes cluster on minikube on Docker. The logs generated are stored using the Splunk log management system.
3. IDS and vulnerability detection - The logs generated are fed into the IDS or vulnerability detector in Ansible for further analysis, where IDS finds potential vulnerabilities and generates a report of the analysis performed. We use the IDS named Snort employed by Ansible for vulnerability detection.

4. Response Mechanism - The list of vulnerabilities will be then sent to Ansible to start the mitigation automation for them. Before this, a ticket will be created for the list of vulnerabilities for each network device.
5. Mitigation Automation - For each device in the network we will automate a particular mitigation mechanism depending on the vulnerability in question. This mitigation will finally make modifications to the appropriate device in the virtual network to resolve the vulnerabilities. This mitigation will be automated using AWX Ansible. After the mitigation automation, a status flag(success or failure) will be sent to the user depicting the success or failure of vulnerability mitigation.

Data Management Plan

The data for this project are mainly the logs generated for each of the devices in the network. The other data generated are the vulnerabilities detected by the IDS (Snort in AWX Ansible) for a particular network device. The tickets created and sent to the users when a threat has been detected can also be considered as data that needs to be managed. We will be configuring a logs management system (Splunk, Kibana, etc.) to facilitate the storage of the logs. To help preserve the data, we document the logs, list of vulnerabilities and the ticket information into three separate files. This will also help us understand the application we run on the virtual network and help in recreating it. Furthermore, we will also be including the network configuration and the application in the documentation. The data will be open source and will be uploaded on GitHub.

Deliverables

1. Project: Automate security scanning for an application, manage incident reports and send alert reports to the assigned leads.
2. Project Report: Document the work done on the automation of security mechanisms, the configuration of new IDP rules and intimating the assigned leads based on network.
3. Weekly Updates: Updating the Technical director with weekly progress reports every Friday.

Issues

Understanding automation in a cloud environment along with log data capture and IDS presents the following challenges:

1. Environment Setup: In order to integrate multiple devices like Routers, Switches and Servers together in a virtual environment, knowledge about networking systems and their security issues are required. In our case, we as a team have strong experience with servers but need to research and analyze routers' and switches' vulnerabilities.
2. Financial: Many of the tools that we used for data capture and alerting are not open source and require funding for the features of this project. Hence, we will either look for open source alternatives for the same or similar workarounds.
3. Fine-tuning existing firewall policies and IDS rules: There is a pre-existing limited set of firewall policies and IDS rules, but these would not be helpful if new network scenarios arise which introduce new novel vulnerabilities. Thus we need to fine-tune the existing rules and policies to accommodate the new vulnerabilities.
4. Time: This project has been assigned a timeline of 12 weeks. Limiting the duration of the project to 12 weeks impacts the scope of the project, and increases the likelihood that deviations to the project plan could cause a major impact on project deliverables.

Bibliography

1. https://www.ansible.com/use-cases/security-automation - Security Automation mechanisms in Ansible
2. https://www.networkworld.com/article/3121275/explaining-security-automation-and-its-evolving-definitions.html - Security Automation and its evolution
3. "Security Automation and Threat Information-sharing Options", Panos Kampanakis

Biographical sketches of the team members

All of the team members have industry experience and therefore have acquired knowledge of one or more scripting languages and have also developed the back-end side of applications using well-known programming languages like Python, C, Java, C++, Go and Rust. The team members also have experience with Ansible configuration like using YAML for setting up jobs in a deployment pipeline.

Due to industrial experience in software companies, all of us are also familiar with the internals of what goes into the compilation and execution of a program. Each of us has also done projects based on operating systems and computer networks during our undergraduate studies which will help us with the research and implementation of this project.

Our team members are well versed with Security and CyberSecurity concepts as well. In addition, our team members have experience working with DevOps tools similar to Ansible like JIRA and Jenkins for continuous integration and deployment. Lastly, we have worked in virtual environments and would therefore not have the added overload of learning to set up a virtual environment from scratch.

Schedule

Tasks	Start Date	End Date	Duration
Perform background research	01/20/2022	01/31/2022	11
Research IDS and firewall applications available	01/23/2022	01/31/2022	8
Build a virtual network on a cloud platform and set up AWX Ansible.	02/01/2022	02/06/2022	5
Install servers, routers and switches on the cloud platform and connect them	02/06/2022	02/10/2022	4
Run python script to generate logs for each network device on Ansible(for multiple networks)	02/10/2022	02/13/2022	3
Use existing IDS and FW policies/ create new ones if required	02/12/2022	02/27/2022	15
Integrate IDS with data management applications	02/26/2022	03/01/2022	3
Build an alerting system based on the data	03/01/2022	03/06/2022	5

Budget

Item	Unit	Cost/Unit	Total
Team Hours	5 members, 12 weeks	$4813	$24065
Fringe Benefits	5 members	$63	$315
Conference Travel	5 members	$1250	$6250
Indirect Costs	1	$17152.8	$17152.8
TOTAL			**$47,782.8**

Broader Impact

As the need for fortified security measures increase, it is hard to keep track of each and every intrusion attack possible. Hence, this project can serve as an automation tool by patching in new updates to the IDS by creating new firewall policies and rulesets for their alert and ticketing systems. This project can also be extended further to provide immediate solutions to the detected threats, thus curbing down the impact and improving the security of the application. A stabilized version of the project can be made available for immediate use to detect vulnerabilities in networks and network devices.

Appendix

Foundations and Applications of Self* Systems (FAS*W) is the conference where we aim to present and publish our paper. The title of the paper that closely resembles our project in the conference is “Cloud Security Automation Framework”. The DOI for the same is https://doi.org/10.1109/FAS-W.2017.164