Web-Scraping Project Report

Adam Steffgen & Ardian Bakii

IT 360 SIEM & Forensics

Sean Sanders

Date of Submission:

Introduction

For this project, our goal was to build a web-scraping tool to gather information from a website including domains (malicious and non-malicious), hashes, IP addresses, URLs, emails, and images. Our tool is based off of a Python script that will return this information back and check which domains are malicious by using the VirusTotal API key. Then it saves all of the output for the script into a text file. There is a second Python script that can take the URL of an image from a website and save the image to be used later on for evidence. All of these components can be used together to help identify possible indicators of compromise and malicious activity.

What Didn't Work

Our initial plan was to use Volatility 3 to extract URLs, cookies, and browsing history from different websites. We worked our way through the installation process for Volatility 3 and ran into some road blocks. We couldn't get the packages to properly install for Volatility 3, even after trying multiple different installation methods. Another issue we came across was trying to find a way to create a memory dump on our Linux VMs. All of the tutorials to install Volatility 3 were Windows-specific which made things more difficult in the installation process. We also had some problems scraping certain websites. Text files and more simple websites were easier to scrape.

What Did Work

We shifted our focus to a new tool called BeautifulSoup for web scraping. There was more documentation and tutorials available to help us install and use this tool. The installation process went much smoother and we were able to get the required packages. With this tool, we were able to extract different indicators of compromise from a website to be used for digital forensic analysis later on. After obtaining this information, we were able to do domain and hash lookups on VirusTotal to identify if they were malicious. We were able to upload a tutorial along with corresponding files on how to recreate this project for public use.

What We Learned

From this project, we learned how to install Python libraries, build Python scripts, how to efficiently extract information from websites to aid in identifying malicious activity. We learned how to output all of this information into a project report and automate the process of using threat intelligence tools to identify confirmed malicious indicators of compromise. We also learned how to use GitHub and create a step-by-step tutorial on how to recreate this project.

Code Documentation

Directions: https://github.com/aasteff/IT360_Project/blob/main/README.md

https://github.com/aasteff/IT360_Project/blob/main/ioc.py

https://github.com/aasteff/IT360_Project/blob/main/save_image.py

https://github.com/aasteff/IT360_Project

Code Explanation

The first script we will look at is the ioc.py. This code asks the user to input an URL to scan for the IOCs. The script then grabs any IP, hash, domain, email, URL, and image URL patterns. Then a VIrusTotal lookup is done on the domains and hashes. It has a limit of 3 per type and has a 15 sec interval between lookup dies to the free version of VirusTotal. Then it takes the output of the IOCs and the VirusTotal lookup and displays them in a human readable format on a text document.

The second script we will look at is save_image.py. This script asks the user to enter an image URL. After the user enters the URL it sends a GET request to download and save the image for the user. This is so the user can save evidence for the future.

Code Start Up

To set up this project follow the README.md on GitHub.