Function Specification

Student: Sean O'Connor – C00224424

Supervisor: Richard Butler

Cybercrime & I.T. Security – CW_KCCYB_B

Institute of Technology Carlow

Contents

Introduction

For this project I am testing the extendibility of the function off the Burp Suite via Java, Python and Ruby. In the research manual, I discussed the current available functions in the Burp Suite before any extending is done and I also performed an analysis and comparison of the 3 languages before selecting Ruby as the language we will be using going forward. Finally, I briefly discussed what functions I believed should be added to the Burp Suite and what they shall do. I intent to add 4 fully working functions into the Burp Suite as extensions using the Ruby language. If successful, the user should be able to simply load the extension file under the extensions tab in the Suite and use them for whatever they need. Below you will find the list of functions I wish to add, including their metrics and their use case diagrams. The reason for this is because not all functions that are currently available in the Burp Suite so I am testing to see how a user may extend the functionalities of the Burp Suite.

Overview

This project will test how well a user can extend the functionality of the Burp Suite. It will add a ping function, which will ping a host and return a result, it will add a ping sweep function, which will perform a ping sweep on a range of hosts, it will add a threaded port scan, which will allow a user a user to perform a threaded port scan on a host, a subdomain crawler which will scan for all subdomains of a target domain, and a web scraper.

Functions to Be Added

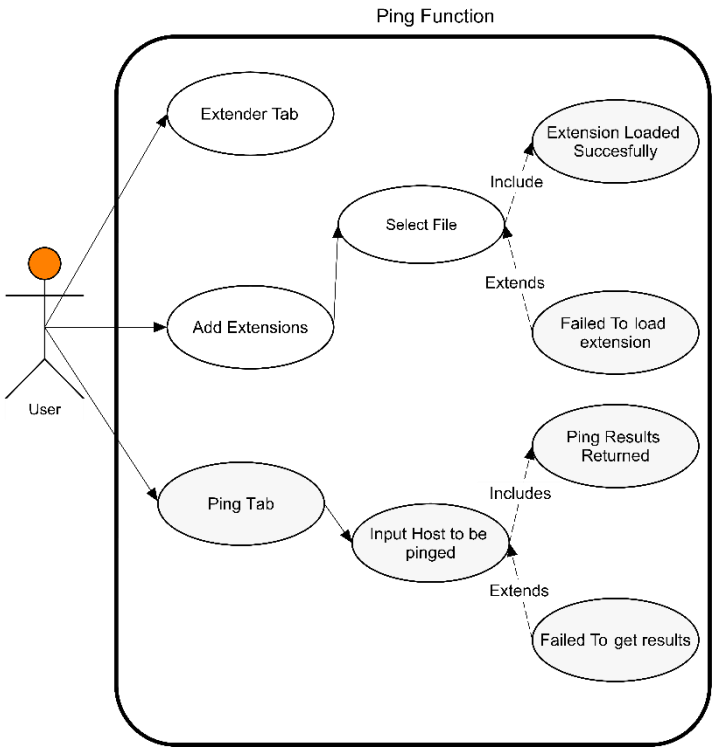
Basic Ping

The first function I would like to add is a simple but necessary function that I am surprised is not already embedded within the Burp Suite itself, a Basic Ping. As the name suggests the Ping function would allow a user to enter in a host name or IP a ping it to see if it is up and live. The layout of the function will be as simple as the actual function itself, consisting of an input field, a button to start the ping and options below for what is returned to the user.

Metrics

- The minimum I would be looking for is for the Ping function to return 4 simple Up/Down responses from a host.
- The next step up from this would be the ping function returning 4 more detailed ping messages, like you would see in a command prompt ping.
- To say this function is fully successful is the function allowing the user to select how many messages it returns and what it returns in the message.

Use Case



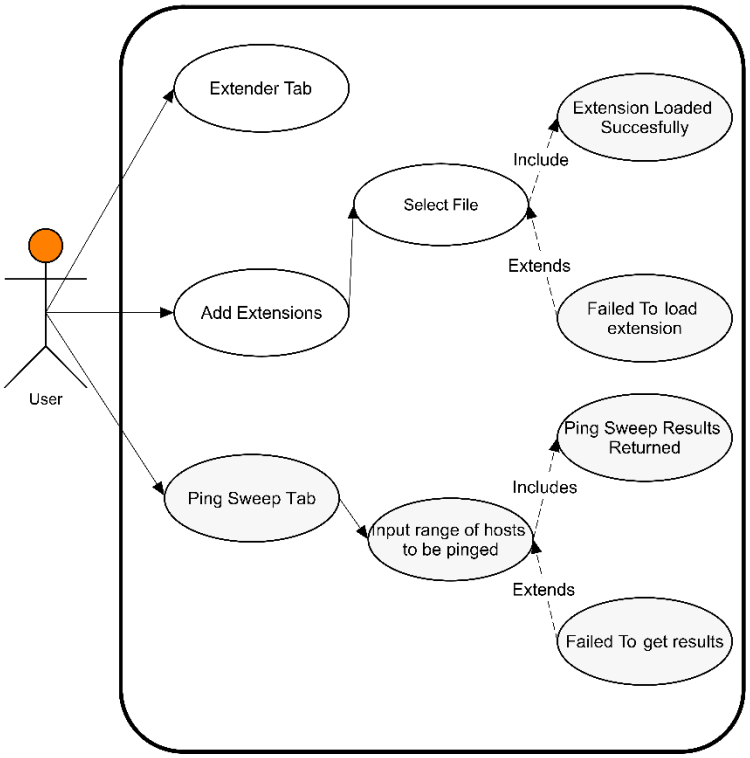
Ping Sweep

The second function I wish to add is not as common as a basic Ping but still useful in a testing scenario. While a basic ping is quick and accurate for a pinging a host, if you are testing a range of hosts it can be time consuming to ping each of these hosts individually so the ideal way of pinging this range is to use a ping sweep. A ping sweep works by instead of pinging one host at a time, it will ping one host and move onto the next before the results come in. This way the sweep moves a lot faster through the range of hosts.

Metrics

- The minimum I would be looking for is a sweep of the full range from 1 to 255 with a count of how many are up and how many are down.
- The next step to make this feature fully functional is to allow the user to select the range they wish to sweep.

Use Case



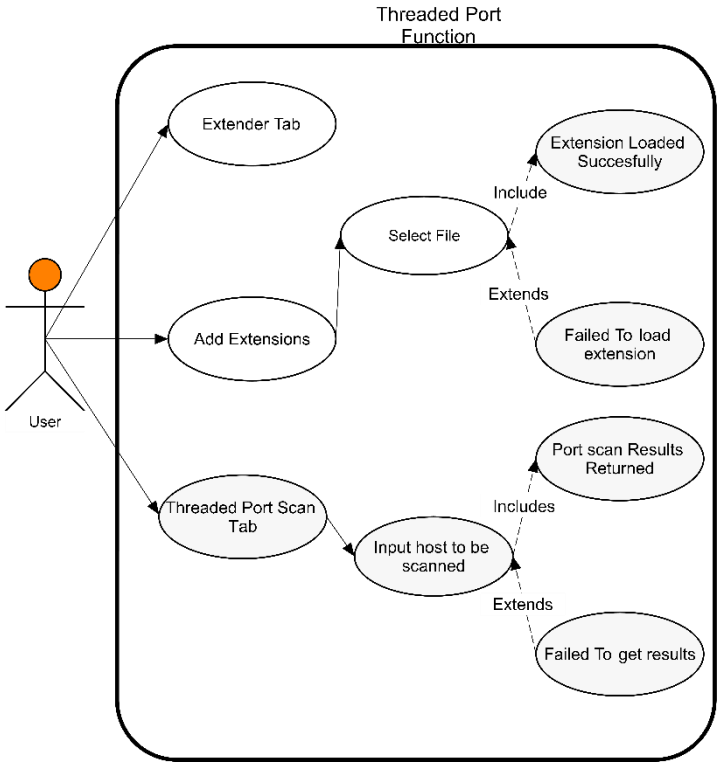
Threaded Port Scan

When a user is testing, whether for reconnaissance purposes or for penetration testing purposes, chances are the user will need to perform a port scan. The purpose of a port scan is to passively scan the ports available on a target host, see which ones are up and which are down, and see which are open and which are closed. While port scans can be very useful for gaining this information, the scan itself can take a very long time. The reason for this is because the scanner will scan each port individually and will wait until it receives a result before moving onto the next port. This is highly inefficient and if scanning a large host, this can take up too much time. So that is why threaded port scanners are used. Threaded port scanners were created with the sole idea of efficiency in mind. Instead of scanning a port and waiting for a response before going onto the next, a threaded port scan will scan a port and while waiting on the result of that port, will move onto the next port, and will follow this pattern until all ports are scanned. This greatly increases the speed of scanning and is a much more efficient method. To create this, I resorted to following a tutorial on geeksforgeeks.org and recreating the code but changing the tutorial to what I needed for my function. (ashishguru9803, 2020)

Metrics

- The minimum I would be looking for is for the threaded port scan to return a count of how many ports are open.
- The next step up from this would be the threaded port scan to return both the open port count and the time taken.
- To say this function is fully successful is the function returning a full list of open ports instead of a count alongside the time taken.

Use Case



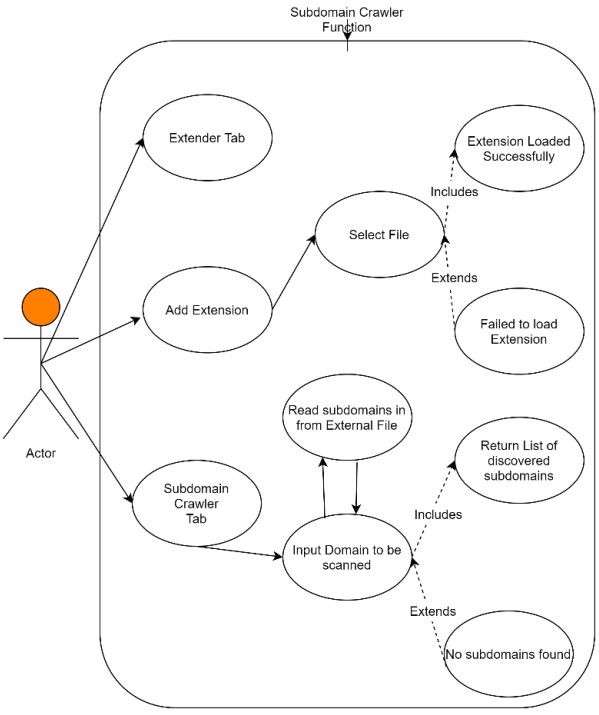
Subdomain Crawler

When a user is performing reconnaissance on a domain, one thing they would need to do is start mapping out the domain itself. We have two functions already that will support this. The ping sweep for finding Ips and the port scanner for identifying open ports. The subdomain crawler is the next step in the level of reconnaissance is to map out the subdomains that exist within the target domain. While the user could individually visit each subdomain or perform an NSlookup on each IP address found in the ping sweep, the easiest method is by using a subdomain crawler. The crawler will allow the user to enter in the name of a domain and perform a scan for the potential subdomains that exist by scanning through a list of potential subdomains and appending them onto the domain that was entered. If the scanner gets a 200 request from the scanned subdomain, it will then know that the subdomain exists, and it will add it to a list which will get returned to the users once the list has been fully scanned through. While this function may be slow to operate, it automates an otherwise tedious job. To create this function, I resorted to a tutorial on thepythoncode.com and recreated the code from the tutorial with changes made to suit the needs of the project. (Rockikz, 2020)

Metrics

- The minimum I would look for is to Have the function only scan for a domain already hard coded into the functions code.
- After this is working at a basic level, I would then want to allow users to enter in a different domain each time so they can perform a wider range of scans for subdomains.
- Once users can enter their own domain to be scanned, I would prefer to allow users to choose which file they wish to use for the scan, the 100-long list, 1000 long or the 10000 long. This can change the level of results and how long the scan takes.
- A final step for the implementation would be to Implement a threader that will Speed up the scan as the process of the scan can take up a lot of the user's time.
- Another extra bit of functionality that could be added is a type of NSlookup functionality built in that gets the IP address of a discovered subdomain.

Use Case



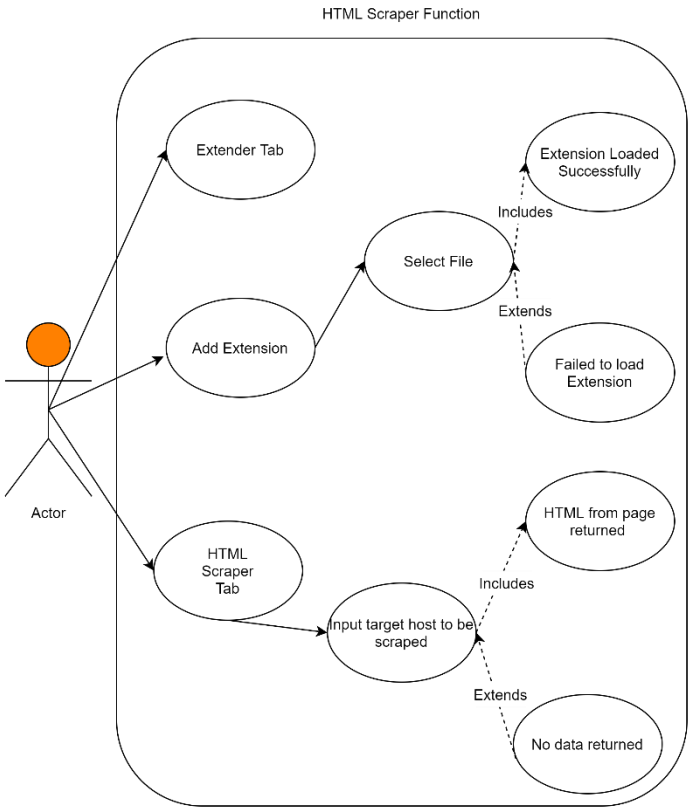
HTML Scraper

A final simple yet useful tool that I attempted to implement was a basic HTML Web scraper. While not an essential tool for reconnaissance, it could still be useful for gathering information and the general layout of web pages, without having to go to the pages and manually get the HTML. The function will allow a user to enter in a domain name and scrape the HTML from that page. This function could potentially work alongside the subdomain scanner, by scraping the HTML from a subdomain whenever one is found, however this functionality does not exist in this extension.

Metrics

- Have the function return the HTML data from a webpage.
- Implement the use the robots.txt for scraping.
- Link the subdomain crawler with the web scraper, so they work together.

Use Case



Bibliography

ashishguru9803, 2020. *Threaded Port Scanner using Sockets in Python.* [Online]
Available at: https://www.geeksforgeeks.org/threaded-port-scanner-using-sockets-in-python/
[Accessed 16 February 2021].

Rockikz, A., 2020. *How to Make a Subdomain Scanner in Python.* [Online]
Available at: https://www.thepythoncode.com/article/make-subdomain-scanner-python
[Accessed 5 April 2021].