Reconnaissance & Scanning

Version Date: 24 SEP 2018

Student Guide Printable Format

Skills and Objectives

Section 7.1: Network Scanning and Reconnaissance

Table of Contents

Facilitation: Open Source Intelligence Gathering

Outcome:

- Understand and be able to gather and document OSINT in relation to operational tasks and targeting.

OSINT Documentation

Introduction
Documenting OSINT collection is a vital part of military operations. Whether you are preparing for offensive operations and gathering specifics concerning a target, or working to defend a network through identifying everything about identified and suspected threats, you will be required to utilize OSINT.

Discussion

What is OSINT (Open Source Intelligence)?

According to SANS:

Open Source Intelligence (OSINT) refers to a collection of data from public sources to be used in an intelligence context, and this type of information is often missed by link crawling search engines such as Google.

According to DoD:

OSINT is "produced from publicly available information that is collected, exploited, and disseminated in a timely manner to an appropriate audience for addressing a specific intelligence requirement."

Why is it important to document your collections, and what should you document?
Documentation is vital because it enables effective operations, reporting, and sharing of critical data. During operations it is key that you only perform actions on identified targets, to limit the exposure of TTP's that were utilized. This also allows for easy collaboration, as documented collection can be shared and spread load for targeted actions by various teams.

What types of artifacts/deliverables do we want to collect and provide to operators?
Examples include:

- Screenshots
- Timelines related to Data
- Sensitive Target Data
- Artifacts/Attributes from target
- Operational Notes and Methods of Retrieval

- Visualization of Collected Data
- Database of Collected Indicators

What can we do to help organize our collected data to enable trending, and easier identification of pertinent information?

- Mind Map to Store and Visual Data Collected
  Examples include:
- OSINTTOOLS GitHub
- OSINT Framework (Online Tools List)
- OSINT Framework GitHub
- Free Mind Map

Allows for exporting, publishing and sharing in a variety of formats.

Collection and Use

Introduction
Gathering OSINT is a vital part of military operations. Whether you are preparing for offensive operations and gathering specifics concerning a target, or working to defend a network through identifying everything about identified and suspected threats, you will be required to perform OSINT collection at various levels within your Branch.

Discussion

What type of information can you find within the public domain?
(Listed below are some of the main cetgories and data types, this can help students gain greater familiarization and understanding)

Web Data Collection

- Cached Content
- Analytics
- Proxy Web Application
- Command Line Interrogation

`Methods:` Burpsuite, OWASP Zap, TheHarvester

Social Media Collection

- Twitter

- Facebook
- Instagram
- People Searches
- Registry and Wish Lists

`Methods:` Family Trees, Birth/Death Records, Content Analysis, Graphic Searches, Open API's

Sensitive Data Collection

- Business Data
- Profiles
- Non-Profits/Charities
- Business Filings
- Historical and Public Listings

`Methods:` Google Maps, Job Postings, Company Public Websites/Portals

Publicly Accessible Data

- Physical Addresses
- Phone Numbers
- Email Addresses
- User Names
- Search Engine Data
- Web and Traffic Cameras
- Wireless Access Point Data

`Methods:` Reverse Lookup, Newsletters, Real Estate Data, Standardized Email Formatting, Google Dorks

Domain and IP Address Data Collection

- DNS Registration
- IP Address Assignments
- Geolocation Data
- Whois

`Methods:` Whois, Traceroute, Web Registrars

What type of operational impact does this data have or how can it be used?
Examples include:

- Technical Analysis/Exploitation
- Phishing Campaigns
- Attacking Web Portals
- Test Common Credentials
- Enumerate Public Facing Targets

DEMO: Google Dorks

NOTE	Reference the Google Hacking DB for examples of usage of syntax and examples Google Hacking Database to show various items that you can collect via Google Search.

`site:``<DOMAIN NAME>`

- Will return website on following domain

`allintitle:``<SPECIFIED PHRASE>`
`intitle:``<SPECIFIED PHRASE>`

- Contains a specified phrase that appears in a title tag on a page

`inurl:``<SPECIFIED PHRASE>`

- Restricts the results contained in the URLS to the specified phrase

`filetype:``<FILE TYPE>`

- Searches for specified file type formats.

SPECIAL CHARACTERS:

`Double apexes "search string"`

- Double quotes are used to search for a specific word or a set of words written exactly that way. `"gelato"`

`Minus sign [ -string ]`

- The minus sign in front of the word tells Google to exclude that specific word from the search. `-cripto`

`Tilde [ ~string ]`

- Specifically, it indicates a Google to search its synonyms over a word.
  `~car`

Operator "OR" | [string OR string_2]

- This logical operator tells Google to look for a word or the other. It can also be used more than once.
  `[monero | serhack]`

HyperText Markup Language (HTML):

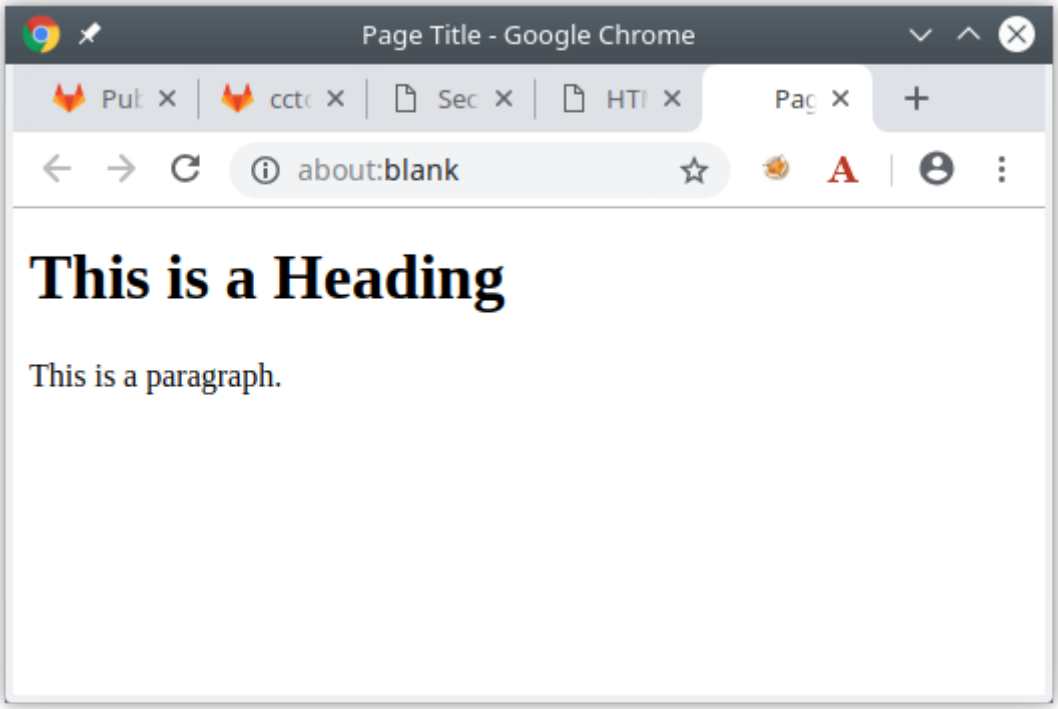
HTML is a domain-specific programming language that defines the structure for data or text to be displayed by a web browser. Some servers contain flat text files of HTML that are returned to the user based on the requested URI, while others generate HTML dynamically based on templates. In either case, the server returns HTML in the request body as a response whatever context is provided by the HTTP request.

HTML standards are recommended by an international community known as the W3C, or the World-Wide Web Consortium. Web browsers then implement the HTML recommendations to display content to the user. Web browsers must then choose to conform to or diverge from the recommendations. For example, historically Internet Explorer was infamous for creating its own features in its web browser that did not exist in the standard. Consequently, websites designed for Internet Explorer displayed differently than in other browsers. More recently, however, most modern browsers conform to the W3C's recommended standards in a mostly uniform fashion.

An simple HTML example page looks like such:

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
</body>
</html>
```

The above source would render as:



HTML operates off of elements through tags such as <body> which can be nested. A tag should be closed such as </body>, most modern system will still display HTML correctly without a closed tag. Not all tags require a closed tag such as img, hr or br, these are know as void elements.

An important note about HTML is that it is designed to dovetail with Cascading Stylesheets (CSS) and Javascript. When a browser loads a web page, it creates a Document Object Model (DOM) of the page. The DOM is a tree of objects based on the HTML source. Javascript can then directly manipulate objects in the DOM, while CSS can style objects.

Instructor should open up the w3schools page on HTML and HTML DOM to show examples of simple HTML and the DOM structure.

DEMO: Simple HTML page

IMPORTANT	This should be done from either Firefox on your workstation, or Chrome on your OPS station. Developer console is disabled in Chrome on your workstation!

1. Browse to Demo-Web_Exploit_upload instance in the DEMO net: http://<ip>/webexample/htmldemo.html
2. Utilize Dev Console (`F12` and select Inspector) or "View Page Source" and discuss the various elements

Sources:

- https://www.w3.org/TR/html51/ - W3C recommendations for HTML 5.1
- https://www.w3schools.com/html/default.asp - W3 Schools HTML tutorial and reference guide

- https://www.w3schools.com/js/js_htmldom.asp - W3 Schools reference guide for HTML DOM

Cascading Stylesheets (CSS)

CSS is a language that describes the style of an HTML document. Specifically, the language defines how a browser should display the HTML DOM. CSS can be stored in-line in a sheet, per object via the style tag, with Javascript, or externally in CSS files. Through external CSS files, the entire theme in a web page can be changed just by modifying a single file.

A sample CSS code block:

```
body {
    background-color: lightblue;
}
h1 {
    color: white;
    text-align: center;
}
p {
    font-family: verdana;
    font-size: 20px;
}
```

More information about CSS can be found at https://www.w3schools.com/css/css_intro.asp

DEMO: Scrapping with Python (lxml)

- Navigate to http://quotes.toscrape.com and demonstrate how to view the website coding via to F12, Developer Menu in order to identify the data we want to scrape.
  - Provides an extensive library written for parsing XML and HTML documents very quickly
- Run commands and retrieving output.

installs modules to be used by Python

```
apt install python3-pip
pip3 install lxml
pip3 install requests
```

Below is the scripting content, each command can be run seperatly within python session as well

```
Unresolved directive in lesson-2-recon_sg.adoc - include::example$scrape.py[]
```

Output of Script/Commands

```
Authors: ['Albert Einstein', 'J.K. Rowling', 'Albert Einstein', 'Jane Austen', 'Marilyn Monroe', 'Albert Einstein', u□Andr\xe9 Gide', 'Thomas A. Edison', 'Eleanor Roosevelt', 'Steve Martin']
```

DEMO: Scrapping with PowerShell (Invoke-Webrequest)

This method provide greater flexibility in how you target data, and lets you do basic PowerShell conditional statements.

Below is the scripting content, each command can be run seperatly within python session as well

```
$url = invoke-webrequest http://quotes.toscrape.com

$authors = ($url.AllElements | ?{$_.class -match 'author'}).innerText

$quotes = ($url.AllElements | ?{$_.class -match 'text'}).innerText
```

Output of Script/Commands

```
PS C:\Users\administrator1> $authors
Albert Einstein
J.K. Rowling
Albert Einstein
Jane Austen
Marilyn Monroe
Albert Einstein
André Gide
Thomas A. Edison
Eleanor Roosevelt
Steve Martin

PS C:\Users\administrator1> $quotes
□The world as we have created it is a process of our thinking. It cannot be changed
without changing our thinking.□
□It is our choices, Harry, that show what we truly are, far more than our abilities.□
□There are only two ways to live your life. One is as though nothing is a miracle. The
other is as though everything is a miracle.□
□The person, be it gentleman or lady, who has not pleasure in a good novel, must be
intolerably stupid.□
□Imperfection is beauty, madness is genius and it's better to be absolutely ridiculous
than absolutely boring.□
□Try not to become a man of success. Rather become a man of value.□
□It is better to be hated for what you are than to be loved for what you are not.□
□I have not failed. I've just found 10,000 ways that won't work.□
□A woman is like a tea bag; you never know how strong it is until it's in hot water.□
□A day without sunshine is like, you know, night.□
Quotes by: GoodReads.com
```

DEMO: Native/Alternate Commands

Web Browser Developer Console (F12)

Other Means identified during earlier in the Network Module: Network Reconnaissance Lesson

Review: Network Reconnaissance

Network Reconnaissance is the process of discovery and enumeration of hosts on a targeted network. The purpose of this is to determine how each host can be leveraged to accomplish a mission. It is iterative process that begins every time access to a new host is gained.

It is broken into three stages which are not dependent on each other.

Host Discovery

1. Determine if hosts exists on the network using **quick** port agnostic scans.

Ping Sweep
Sends one icmp echo request packet to each host on the `192.168.1.0/24`

- Linux: `for i in {1..254} ;do (ping -c 1 192.168.1.$i | grep "bytes from" &) ;done`
- Windows: `for /L %i in (1,1,255) do @ping -n 1 -w 200 192.168.1.%i > nul && echo 192.168.1.%i is up.`

Port Enumeration

2. Determine what ports on a target machine are available to be communicated with. *These ports indicate which services are potentially listening on a target machine and are not blocked by a network or host security appliance.

Use nmap to scan a range and specific ports on a discovered machine:

- `nmap -sS -Pn 8.8.8.8 -p 135-139,22,80,443,21,8080`

NOTE	Nmap requires the `-Pn` switch to be run over Proxychains. It requires this switch because proxychains does not support `ICMP` traffic. Nmap pings each target first before attempting to enumerate selected ports.

Use **nc** to scan a range and specific ports on a discovered machine:

- `nc -z -v -w 1 8.8.8.8 440-443`

NOTE	`nc` does not ping an IP Address before enumerating the ports; therefore, it works great over proxychains.

Port Interrogation

3. Interact with discovered hosts and ports to determine the best way to leverage each available service.

Use nc to interrogate a web server:

- `nc -Cv 127.0.0.1 80`
- Type: `GET / HTTP/1.0` to get a HTTP Response header from the server.

Use nmap to perform service detection on port 22 of your opstation:

- `nmap -sV 127.0.0.1 -p 22`

Using nikto to perform a vulnerability scan on your opstation:

- `nikto -h 127.0.0.1 -p 80`
    - Also shows other information like what HTTP methods are allowed and various CVE vulnerabilities.

Sources:

- Why Proxychains does not support ICMP

Facilitation: Nmap Scripting Engine (NSE) Scripts

Outcome:
Understand how to perform and utilize advanced scans utilizing Nmap NSE to enhance and develop a accurate penetration test.

Advanced Network Scanning

Introduction
The Nmap Scripting Engine (NSE) is one of Nmap's most powerful and flexible features. It allows users to write (and share) simple scripts to automate a wide variety of networking tasks. Those scripts are then executed in parallel with the speed and efficiency you expect from Nmap.

Users can rely on the growing and diverse set of scripts distributed with Nmap, or write their own to meet custom needs. The core of the Nmap Scripting Engine is an embeddable Lua interpreter. Lua is a lightweight language designed for extensibility.

Network Discovery

- Examples include looking up whois data based on the target domain, querying ARIN, RIPE, or APNIC for the target IP to determine ownership, performing identd lookups on open ports, SNMP queries, and listing available NFS/SMB/RPC shares and services.

Sophisticated Version Detection

- Able to recognize thousands of different services through its probe and regular expression signature based matching system… version detection now calls NSE by default to handle some tricky services.

Vulnerability Detection

- While Nmap isn't a comprehensive vulnerability scanner, NSE is powerful enough to handle even demanding vulnerability checks… Many vulnerability detection scripts are already available.

Backdoor Detection

- Some of these can be detected by Nmap's regular expression based version detection, but more complex worms and backdoors require NSE's advanced capabilities to reliably detect.

Vulnerability Exploitation

- As a general scripting language, NSE can even be used to exploit vulnerabilities rather than just find them.

Script Familiarization

Scripts are stored in a `scripts` subdirectory of the Nmap data directory by default:

`/usr/share/nmap/scripts`

For efficiency, scripts are indexed in a database stored in

`../scripts/script.db`

DEMO: NSE Familiarization

— Navigate to Nmap NSE `scripts` folder and view script.

— View the Nmap NSE repository online, and see how to download what you want.

— Identify the various components within the script in comparison with where they show when you run a scan.

The following are code snippets from the `banner.nse` script.

SCRIPT CONTENTS

- Provides a basic description of its function
- Provides a sample output of what is expected when the script is run
- Provides general info (Author, Category, Etc.)

The remainder of the script defines the arguments and functions utilized by the script to perform a banner grab

```
Unresolved directive in lesson-2-recon_sg.adoc - include::example$nse-example.lua[]
```

USAGE AND EXAMPLES

`nmap --script <filename>|<category>|<directory>|<expression>[,…]`

- Runs all scripts that match defined criteria.

`nmap --script-help "<filename>|<category>|<directory>|<expression>[,…]"`

- Shows help content for specific scripts, categories, etc.

`nmap --script-args <args>`

- Allows options definied within the script to be ran in conjunction with the script.

`nmap --script-args-file <filename>`

- Allows options definied within the script to be pre listed in a file and then ran in conjunction with the script.

`nmap --script-help <filename>|<category>|<directory>|<expression>|all[,…]`

- Shows help about scripts. For each script matching the given specification, Nmap prints the script name, its categories, and its description.

`nmap --script-trace`

- Similar to --packet-trace as it will output traffic data to include protocol, source, destination, and transmitted data.

dns-brute.nse

- Find valid DNS (A) records by trying a list of common sub-domains and finding those that successfully resolve.

```
nmap -p 80 --script dns-brute.nse <domain name>
```

hostmap-bfk.nse

- Find virtual hosts on an IP address that you are attempting to compromise (or assess).

```
nmap -p 80 --script hostmap-bfk.nse <domain name>
```

traceroute-geolocation.nse

- Perform a traceroute to your target IP address and have geolocation data plotted for each hop along the way.

```
nmap --traceroute --script traceroute-geolocation.nse -p 80 <domain name>
```

http-enum.nse

- Attempts will be made to find valid paths on the web server that match a list of known paths for common web applications.

```
nmap --script http-enum <IP Address>

  --script-args http-enum.basepath='<Web Server Dir/>' <IP Address>
```

- This entry shows an example of utilizing the `--script-args` option, identifying a valid value from within the script in order to narrow the scan.

smb-os-discovery.nse

- Determines the operating system, computer name, netbios name and domain of a system.

```
nmap -p 445 --script smb-os-discovery <IP Address / Subnet>
```

firewalk.nse

- Discovers firewall rules using an IP Protocol Time To Live (TTL) expiration technique

```
nmap -p 80 --script=firewalk.nse <IP Address>

  --script-args=firewalk.max-retries=1 <IP Address>
  --script-args=firewalk.probe-timeout=400ms <IP Address>
  --script-args=firewalk.firewalk.max-probed-ports=7 <IP Address>
```

This entry shows examples of utilizing the `--script-args` option, identifying a valid value from within the script in order to narrow the scan.

DEMO: NSE Familiarization

Review and execute some of the scripts outlined above

Sources:

- Nmap scripts library on github
- Nmap Scripting Engine website
- Nmap Scripting Engine Man Page
- Nmap Usage and Examples
- SEC487: Open-Source Intelligence (OSINT) Gathering and Analysis
- A curated list of amazingly awesome OSINT