CSCI 465 – Network/Application Security

Mia Weber

11/06/2023

Web Security Basics – TryHackMe

How the Web Works Module

DNS in Detail

What is DNS?

- DNS (Domain Name System) = A simple way for us to communicate with devices on the internet without remembering complex numbers (IP addresses).

Domain Hierarchy

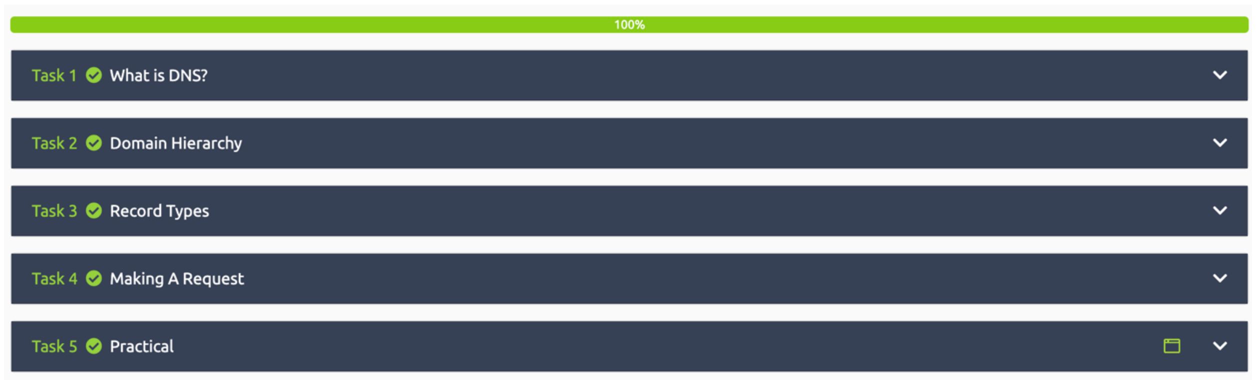
- TLD (Top-Level Domain) = rightmost part of a domain name. For example, tryhackme.com's TLD is .com
- Second-Level Domain = limited to 63 characters. Example in tryhackme.com is tryhackme
- Subdomain = On the left-hand side of the Second-Level Domain using a period to separate it. For example, in admin.tryhackme.com the subdomain is admin. You can use multiple subdomains, but the length must be 253 characters or less. There is no limit to the number of subdomains you can create for your domain name.

Record Types

- A record = IPv4 addresses
- AAAA record = IPv6 addresses
- CNAME record = resolve to another domain name. ex: store.tryhackme.com -> shops.shopify.com
- MX record = email
- TXT record = text or notes

Making A Request

1. Checks local cache. If not a recent lookup a request to the recursive DNS server is made
2. Recursive DNS server checks its local cache. If not found, journey begins to find the correct answer
3. Root servers act as DNS backbone and redirect you to the correct TLD server. Connects you to the correct TLD server with the .com address for example
4. TLD server holds records for where to find authoritative server to answer the DNS request (the nameserver)
5. An authoritative DNS server has the answer, shares back with the Recursive DNS server to update its local cache, and then relayed back to the original client that made the DNS request


100%

Task 1 What is DNS?

Task 2 Domain Hierarchy

Task 3 Record Types

Task 4 Making A Request

Task 5 Practical

The DNS in detail room introduces the topic of Domain Name System (DNS) and walks through how DNS works, what exactly it does and the many different types of DNS records. The room also provided a hands-on exercise with nslookup and detailed information about making DNS requests. Although I have a lot of experience with DNS it is always helpful to get a reminder of the types of DNS records and the how DNS requests are propagated through a network. I can always benefit from learning more about the common things that go wrong with DNS since it often is at the root of a lot of networking configuration issues, so I wish the room had more information about that but overall, it was a good refresher on DNS.

HTTP in Detail

What is HTTP(S)

- HTTP (HyperText Transfer Protocol) = The set of rules used for communicating with web servers for the transmitting of webpage data (HTML, Images, Videos, etc)
- HTTPS = Secure version on HTTP. Data is encrypted

Requests and Responses

- URL (Uniform Resource Locator) = An instruction on how to access a resource on the internet
- Scheme = Instructs on what protocol to use for accessing the resource such as HTTP, HTTPS, FTP
- User = Services that require authentication to log in can have a username and password in the URL to log in
- Port = Port you are going to connect to – usually 80 for HTTP
- Path = File name or location of the resource being accessed
- Query String = Extra bits of info that can be sent to the requested path
- Fragment = Reference to a location on the actual page requested. Commonly used for pages with long content and can have a certain part of the page linked to it.

HTTP Methods

- HTTP Method = A way for the client to show their intended action when making an HTTP request.
- GET request = Used for getting info form a web server
- POST request = Used for submitting data to the web server and potentially creating new records
- PUT request = Used for submitting data to a web server to update info

- DELETE request = Used for deleting info/records from a web server

HTTP Status Codes

- 100-199 Information Response = Sent to tell the client the first part of their request has been accepted and they should continue sending the rest of their request. No longer common
- 200-299 Success = Used to tell the client their request was successful
- 300-399 Redirection = Used to redirect the client's request to another resource. Can be either to a different webpage or website
- 400-499 Client Errors = Used to inform the client there was an error with their request
- 500-599 Server Errors = Reserved for errors happening on the server-side and usually indicate a major problem with the server handling the request
- 200 = ok, 201 = created, 302 = found, 400 = bad request, 404 = page not found, 503 = service unavailable

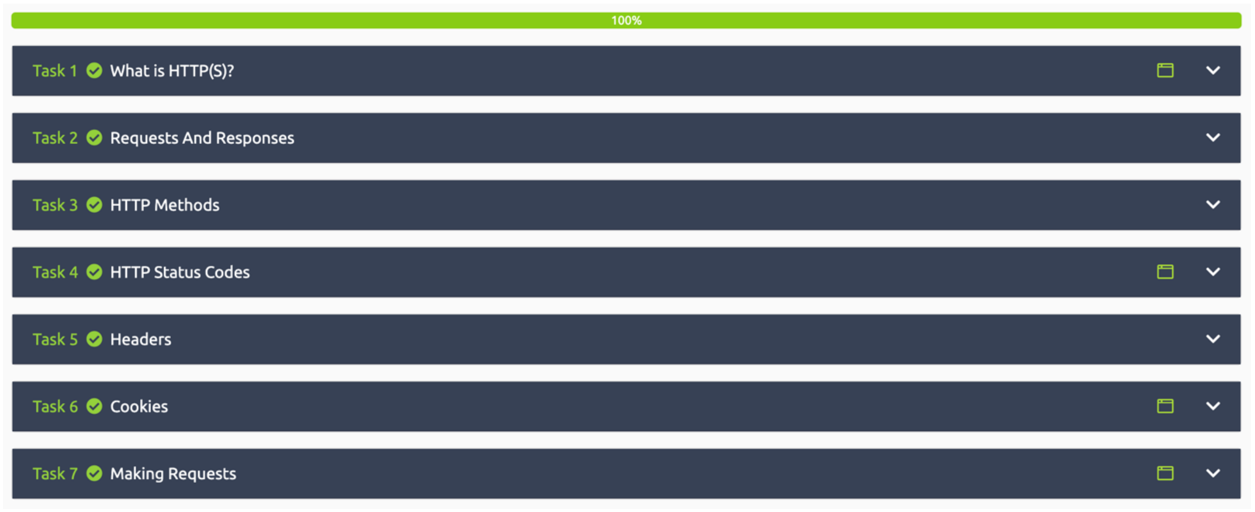
Headers

- Common Request Headers:
- Host = By providing the host headers, you can tell it which website (for web servers that host multiple websites) you require
- User-Agent = Browser software and version number, this helps the web server format the website for your browser
- Content-Length = Tells the web server how much data to expect in the web request so the server can be sure it isn't missing data
- Accept-Encoding = Tells the web server what types of compression methods the browser supports so the data can be made smaller to transmit
- Cookie = Data sent to server to remember your info
- Common Response Headers:
- Set-Cookie = Information to store which gets sent back to the web server on each request
- Cache-Control = How long to store the content of the response in the browser's cache before it requests again
- Content-Type = Tells the client what type of data is being returned
- Content-Encoding = What method has been used to compress the data

Cookies

- Because HTTP is stateless, cookies can be used to remind the web server of who you are, personal settings for the website, and whether you've been to the website before
- Commonly used for website authentication
- Usually won't be a clear-text string but a token

Making Requests


100%

Task 1 What is HTTP(S)?

Task 2 Requests And Responses

Task 3 HTTP Methods

Task 4 HTTP Status Codes

Task 5 Headers

Task 6 Cookies

Task 7 Making Requests

This room outlines HTTP, cookies, status codes, and details HTTP requests and responses and request and response headers. I found the sections on request headers and response headers to be especially helpful reminders. There is a lot to remember about HTTP, so it was good to go through these sections and since a lot of this is memorization it was useful for me to take more detailed notes.

Introduction to Web Hacking Module

Walking an Application

Exploring the Website

- Pen tester's goal is to discover features that are vulnerable and attempt to exploit them. These features are usually parts of the website that require interactivity with the user

Viewing The Page Source

- Page Store = the human-readable code returned to our browser/client from the web server when we make a request
- Comments are marked with <!—and -->
- Links to different pages in HTML are written in anchor tags (<a) and the link you are directed to is stored in *href* attribute
- Many modern websites use a framework as opposed to being made from scratch. Framework is a collection of premade code that allows developers to include common features that a website requires, saving time
- Viewing the page source gives us clues about which framework and what version of that framework – this allows you to determine if it is out of data and housing popular vulnerabilities

Developer Tools – Inspector

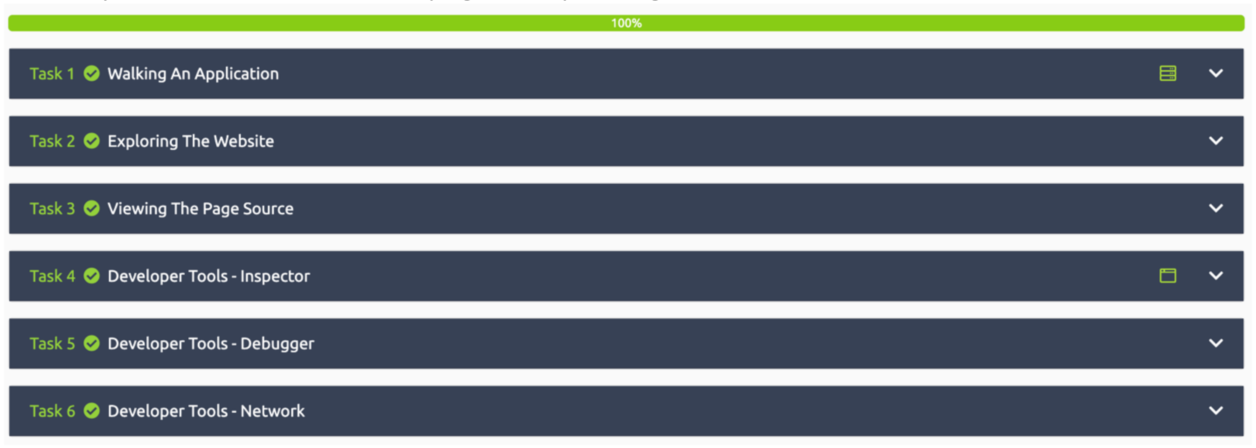
- Modern browser includes developer tools – tool kit used to aid web developers in debugging web applications
- Page source doesn't always represent what's shown on the webpage because CSS, JS, and user interaction can change the content and style of the page. Element inspector provides a live representation of what is currently on the website

Developer Tools – Debugger

- This panel in developer tools is indented for debugging but for pen testers it gives the option of digging deep into the JS code. This feature is sometimes called sources

Developer Tools – Network

- The network tab on the developer tools can be used to keep track of every external request a webpage makes. If you click on the Networks tab and then refresh the page, you'll see all the files the page is requesting


100%

Task 1 Walking An Application

Task 2 Exploring The Website

Task 3 Viewing The Page Source

Task 4 Developer Tools - Inspector

Task 5 Developer Tools - Debugger

Task 6 Developer Tools - Network

This room reviewed the many different tools that exist to help review a web application and search for security issues using browser tools. Built-in tools like Page Source, Inspector, Debugger, and Network were all reviewed and explained in detail. Although I have used all these tools before, it was always to debug a web application and never to discover vulnerabilities or address security concerns. It was helpful to get an in-depth explanation of the tools in the context of their impact on web application security.

Content Discovery

- Three main ways of discovering content on a website: manually, automated, and OSINT (Open-Source Intelligence)

Manual Discovery – Robots.txt

- The robots.txt file is a document that tells search engines which pages they are and aren't allowed to show on their search engine results or ban specific search engines from crawling the website
- It can be common practice to restrict certain website areas so they aren't displayed in search engine results. This file gives us a list of locations on the website that the owners don't want us to discover as pen testers

Manual Discovery – Favicon

- The favicon is a small icon displayed in the browser's address bar or tab
- When frameworks are used to build a website, a favicon that is part of the installation gets leftover, and if the website developer doesn't replace this with a custom one, it can give us a clue on what framework is in use
- Once we know the framework stack, we can use external resources to discover more about it

Manual Discovery – Sitemap.xml

- The sitemap.xml file gives a list of every file the website owner wishes to be listed on a search engine

- These can contain areas of the website that are a bit more difficult to navigate to or even list some old webpages that the current site no longer uses but are still working behind the scenes

Manually Discovery – HTTP Headers

- When we make requests to the web server, the server returns HTTP headers that can contain useful information such as the webserver software and even the programming language being used
- We can use this to identify vulnerable versions of the software being used

Manual Discovery – Framework Stack

- Once you've established the framework of a website you can locate the framework's website and we can learn more about the software and other information

OSINT – Google Hacking / Dorking

- Google hacking / dorking = utilize Google's advanced search engine features which allow you to pick out custom content such as results from a certain domain name using the site.
- Site = Returns results only from the specified website address
- Inurl = Returns results that have the specified word in the URL
- Filetype = Returns results which are a particular file extension
- Intitle = Returns results that contain the specified word in the title

OSINT – Wappalyzer

- Wappalyzer is an online tool and browser extension that helps identify what technologies a website uses such as frameworks, Content Management Systems (CMS), payment processors, and more. It can find versioning also

OSINT – Wayback Machine

- The Wayback Machine is a historical archive of websites that dates back to the late 90s. It can help to uncover old pages that may still be active on the current website

OSINT – GitHub

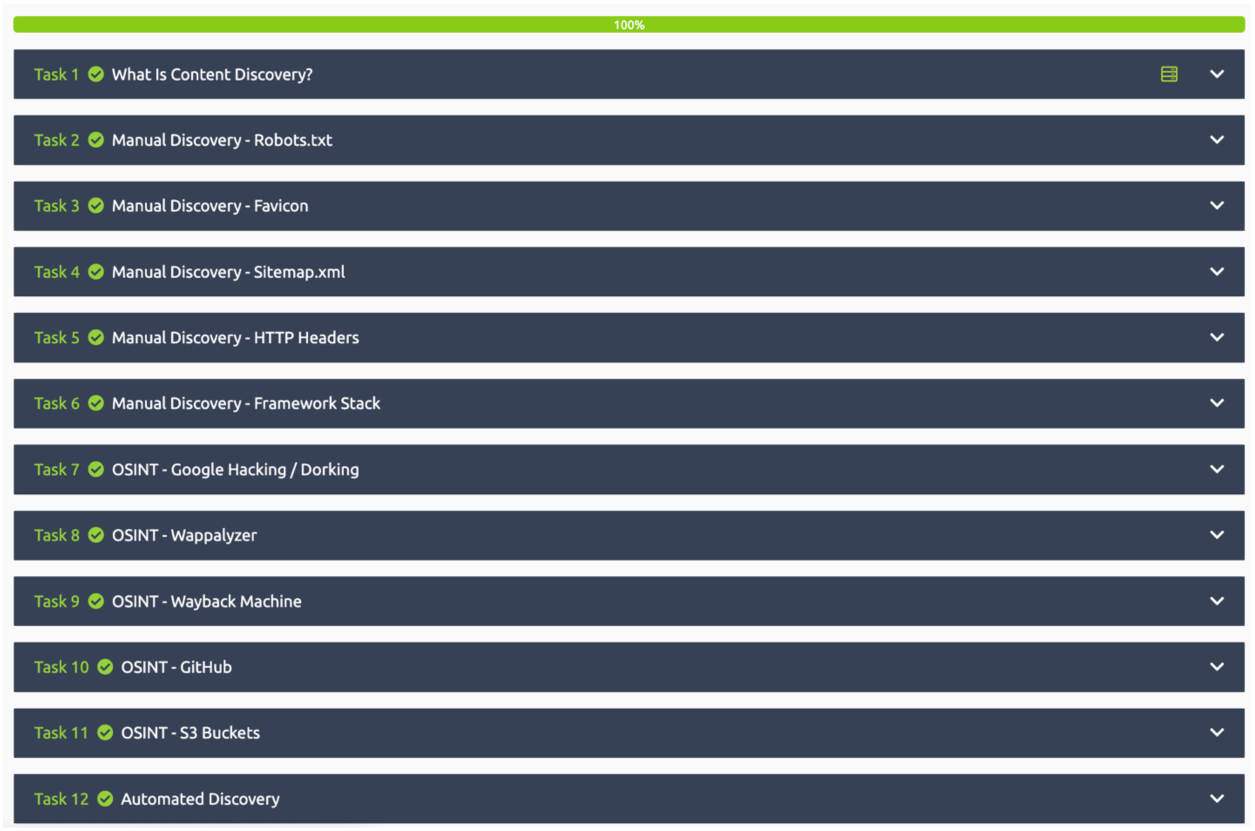
- You can use GitHub's search feature to look for company names or website names to try and locate repositories belonging to the target. If discovered, you might have access to source code, passwords, or other content that you hadn't found yet

OSINT – S3 Buckets

- S3 buckets can be discovered in many ways including finding the URLs in the website page source, GitHub repos, or automating the process
- One common automation method is using the company name followed by common terms such as assets, www, public, private, etc

Automated Discovery

- Automated Discovery = The process of using tools to discover content rather than doing it manually
- These request check whether a file or directory exists on a website, giving us access to resources we didn't know existed before
- Wordlists = Text files that contain a long list of commonly used words and can cover many use cases ex a passwords wordlist
- Three automation tools include: ffuf, dirb, and Gobuster


100%

Task 1 What Is Content Discovery?

Task 2 Manual Discovery - Robots.txt

Task 3 Manual Discovery - Favicon

Task 4 Manual Discovery - Sitemap.xml

Task 5 Manual Discovery - HTTP Headers

Task 6 Manual Discovery - Framework Stack

Task 7 OSINT - Google Hacking / Dorking

Task 8 OSINT - Wappalyzer

Task 9 OSINT - Wayback Machine

Task 10 OSINT - GitHub

Task 11 OSINT - S3 Buckets

Task 12 Automated Discovery

This room went over the different ways that we can discover hidden or forgotten content on a website that could contain vulnerabilities. Parts of a website that still implement old code can be missing important code patches that expose security vulnerabilities. This room covered manual discovery methods for finding content as well as using tools. The information in the manual discovery sections was straightforward but I was largely unfamiliar with the information in the OSINT (Open-Source Intelligence) sections, so I found those particularly helpful to review new methods of content discovery on web applications.

Burp Suite Free Modules

Burp Suite: The Basics

What is Burp Suite

- Burp Suite = Java-based framework designed to serve as a comprehensive solution for conducting web application pen-testing. It is an industry standard tool for hands-on security assessments of web and mobile applications
- Burp Suite captures and enables manipulation of all the HTTP/HTTPS traffic between a browser and a web server
- Burp Suite Professional is an unrestricted version of Burp Suite Community
- Burp Suite Enterprise is primarily utilized for continuous scanning. It has an automated scanner that periodically scans web applications for vulnerabilities, like Nessus. It resides on a server and constantly scans the target web applications for potential vulnerabilities

Features of Burp Community

- Proxy = The Burp Proxy enables interception and modification of requests and responses while interacting with web applications
- Repeater = Allows for capturing, modifying, and resending the same request multiple times. Particularly useful when crafting payloads through trial and error or testing the functionality of an endpoint for vulnerabilities
- Intruder = Allows for spraying endpoints with requests. Commonly utilized for brute-force attacks or fuzzing endpoints
- Decoder = Can decode captured information or encode payloads before sending them to the target
- Comparer = Enables the comparison of two pieces of data at either the word or byte level. The ability to send potentially large data segments directly to a comparison tool with a single keyboard shortcut significantly accelerates the process
- Sequencer = Typically employed when assessing the randomness of tokens, such as session cookie values or other supposedly randomly generated data. If the algorithm used for generating these values lacks secure randomness, it can expose avenues for devastating attacks
- The Burp Suite Extender module allows for quick and easy loading of extensions into the framework and BApp Store enables downloading of third-party modules

Installation

- Burp Suite can be installed on Linux, macOS, and Windows

The Dashboard:

- Tasks = Tasks menu allows you to define background tasks that Burp Suite will perform while you use the application
- Event Log = Event Log provides information about the actions performed by Burp Suite, such as starting the proxy and details about connections made through Burp
- Issue Activity = Specific to Burp Suite Professional. Displays the vulnerabilities identified by the automated scanner
- Advisory = Advisory section provides more detailed information about the identified vulnerabilities, including references and suggested remediations

Navigation

- Module Selection = Top row of menu bar displays available modules
- Sub-Tabs = If a module has multiple sub-tabs, they can be accessed through the second menu bar that appears directly below the main menu bar
- Detaching Tabs = You can detach multiple tabs to view them in separate windows

Options

- Global Settings = These affect the entire Burp Suite installation and are applied every time you start the application. They provide a baseline configuration for your environment
- Project Settings = Specific to the current project and apply only during the session. The Burp Suite community edition does not support saving projects, so any project-specific options will be lost when you close Burp
- Search = Enables searching for specific settings using keywords
- Type Filter = Filters the settings for User and Project options

    - User Settings = Shows settings that affect the entire Burp Suite installation
    - Project Settings = Displays settings specific to the current project
- Categories = Allows selecting settings by category

Introduction to the Burp Proxy

- The Burp Proxy enables the capture of requests and responses between the use and the target web server. This intercepted traffic can be manipulated, sent to the other tools for further processing, or explicitly allowed to continue to its destination
- Intercepting Requests = When requests are made through the Burp Proxy, they are intercepted and held back from reaching the target server
- Taking Control = The ability to intercept requests empowers testers to gain complete control over web traffic, making it invaluable for testing web applications
- Capture and Logging = Burp Suite captures and logs requests made through the proxy by default, even when the interception is off
- WebSocket Support = Burp Suite also captures and logs WebSocket communication, providing additional assistance when analyzing web applications
- Logs and History = Captured requests can be viewed in the HTTP history and WebSockets history sub-tabs, allowing for retrospective analysis

Connecting Through the Proxy (FoxyProxy)

- Important reminders:
    - When the proxy configuration is active and the intercept is switched on in Burp Suite, your browser will hang when you make a request
    - Be cautious not to leave the intercept switched on unintentionally, it can prevent your browser from making any requests
    - Right-clicking on a request in Burp Suite allows you to perform various actions, such as forwarding, dropping, sending to other tools, or selecting options from the right-click menu

Site Map and Issue Definitions

- Three sub-tabs of the Target tab in Burp Suite
    - Site map = Allows us to map out the web applications we are targeting in a tree structure. Every page that we visit while the proxy is active will be displayed on the site map. This feature enables us to automatically generate a site map by simply browsing the web application. Particularly useful for mapping out APIs as any API endpoints accessed by the web application will be captures in the site map
    - Issue Definition = Provides an extensive list of web vulnerabilities, complete with descriptions and references. This resource can be valuable for referencing vulnerabilities in reports or assisting in describing a particular vulnerability that may have been identified during manual testing
    - Scope Settings = Allows us to control the target scope in Burp Suite. Enables us to include or exclude specific domains/IPs to define the scope of our testing

The Burp Suite Browser

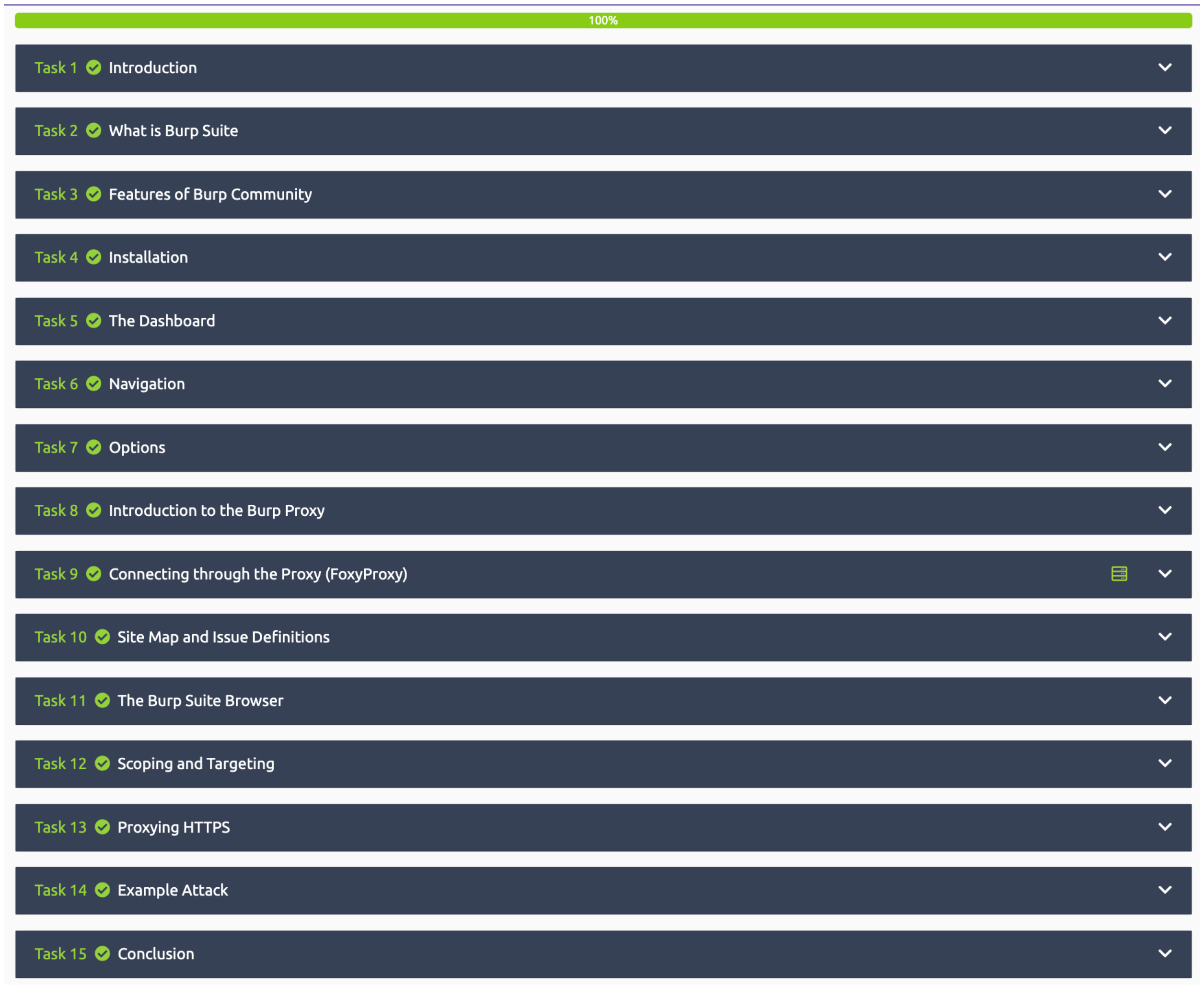
- Burp Suite also includes a built-in Chromium browser that is pre-configured to use the proxy without any of the modifications we just had to do

Scoping and Targeting

- By setting a scope for the project, we can define what gets proxied and logged in Burp Suite
- The Scope settings window allows us to control our target scope by including or excluding domains/Ips

Proxying HTTPS

- When intercepting HTTP traffic, you might encounter an issue when navigating to sites with TLS enabled. To overcome this issue, we can manually add the Port Swigger CA certificate to our browser's list of trusted certificate authorities


100%

Task 1 Introduction
Task 2 What is Burp Suite
Task 3 Features of Burp Community
Task 4 Installation
Task 5 The Dashboard
Task 6 Navigation
Task 7 Options
Task 8 Introduction to the Burp Proxy
Task 9 Connecting through the Proxy (FoxyProxy)
Task 10 Site Map and Issue Definitions
Task 11 The Burp Suite Browser
Task 12 Scoping and Targeting
Task 13 Proxying HTTPS
Task 14 Example Attack
Task 15 Conclusion

This room contained a lot of information about what Burp Suite is and what it can be used for. The sections contained lots of background information and guides on where things are in the tool as well as tasks that require examining an example attack. I found this room especially interesting since I have had no experience with Burp Suite, and it was all new information to me. I found the sections on the features of Burp Suite and the introduction to the Burp Proxy to be the most useful.

Burp Suite: Repeater

What Is Repeater

- Burp Suite Repeater = enables us to modify and resent intercepted requests to a target of our choosing. It allows us to take requests captured in the Burp Proxy and manipulate them, sending them repeatedly as needed
- The ability to edit and resend requests multiple times makes Repeater invaluable for manual exploration and testing of endpoints
- Request List = Displays the list of repeater requests. Multiple requests can be managed simultaneously, and each new request sent to Repeater will appear here
- Request Controls = These controls allow us to send a request, cancel a hanging request, and navigate through the request history
- Request and Response View = We can edit the request in the Request view and then forward it, while the corresponding response will be shown in the Response view
- Layout Options = Enable us to customize the layout of the Request and Response views. The default setting is a side-by-side layout
- Inspector = Allows us to analyze and modify requests in a more intuitive manner than using the raw editor
- Target = Specified the IP address or domain to which the requests are sent. When requests are sent to Repeater from other Burp Suite components, this field is automatically populated

Basic Usage

- Request view will populate when sending a request from the Proxy module to Repeater

Message Analysis Toolbar

- Pretty = Takes the raw response and applies slight formatting enhancements to improve readability
- Raw = Displays the unmodified response directly received from the server without any additional formatting
- Hex = Examine the response in a byte-level representation, which is useful when dealing with binary files
- Render = Allows us to visualize the page as it would appear in a web browser.

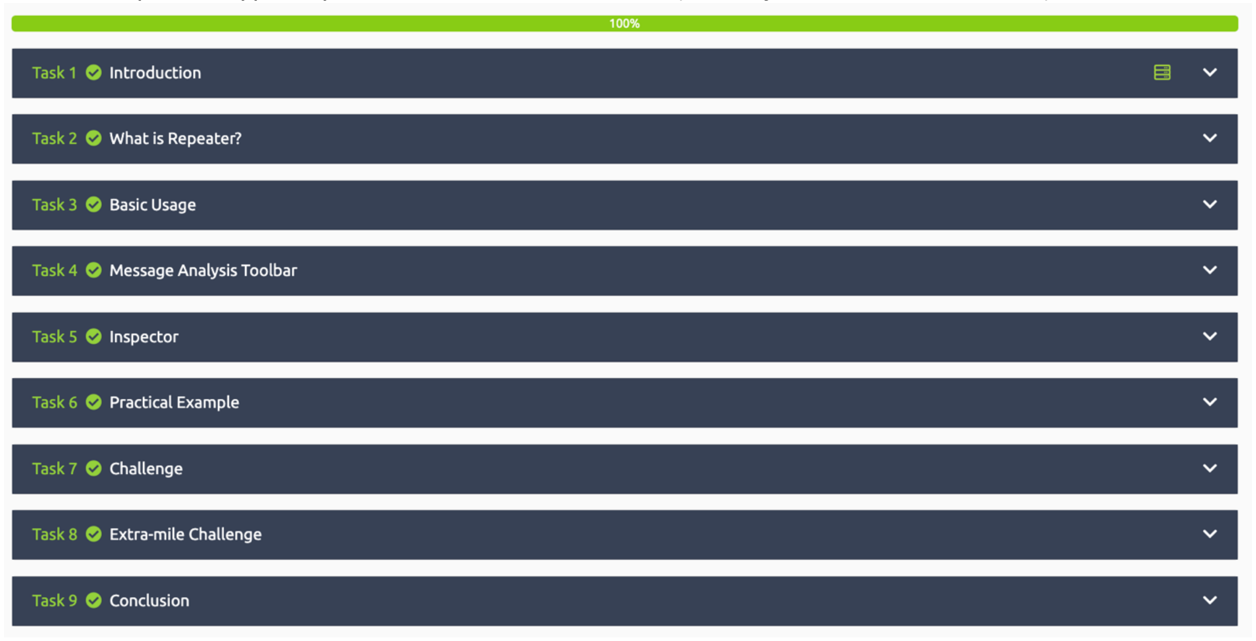
Inspector

- Inspector = Supplementary feature to the Request and Response views in the Repeater module. Used to obtain a visually organized breakdown of request and responses, as well as for experimenting to see how changes made using the higher-level inspector affect the equivalent raw versions
- Inspector can be utilized in both the Proxy and Repeater module
- Request Query Parameters = Refer to data sent to the server via the URL
- Request Body Parameters = Like query parameters but specific to POST requests. Any data sent as part of a POST request will be displayed in this section so we can modify parameters before resending
- Request Cookies = Section contains a modifiable list of cookies sent with each request
- Request Headers = Enables us to view, access, and modify any headers sent with our requests

- Response Headers = Displays the headers returned by the server in response to our request. It cannot be modified, as we have no control over the headers returned by the server. This section becomes visible only after sending an request and receiving a response

Practical Example

- Repeater is particularly well-suited for tasks requiring repetitive sending of similar requests, typically with minor modifications (SQL injection vulnerabilities)



100%

Task 1 Introduction
Task 2 What is Repeater?
Task 3 Basic Usage
Task 4 Message Analysis Toolbar
Task 5 Inspector
Task 6 Practical Example
Task 7 Challenge
Task 8 Extra-mile Challenge
Task 9 Conclusion

This room expands beyond the basics of Burp Suite and explains a lot more details about the Burp Suite Repeater module. The room covers how to use the Repeater module to manipulate and resent captured requests and has a challenge and extra-mile challenge to give hands-on practice to the user. Like the previous room, this room also covered material that I was largely unfamiliar with, and it took me more time to complete the challenge and extra-mile challenge sections. This room did a good job giving experience but also thoroughly explaining the many different security applications of this tool.