**Tenet**

Difficulty: Medium

*Machine: Linux*

**Nmap**

Performing a basic nmap scan shows SSH and a web server are running on the target machine. The website is where I will begin enumerating

| Nmap Scan |
| --- |

**Apache Web Server**

The Apache web server is a skeleton website with the standard apache front page. I need to fuzz to find other directories

Below is the fuzz scan. We see there is a wordpress site open. That is the next best place to look.

| Fuzz scan |
| --- |

Upon investigation of this wordpress site, we see some basic pages. One page seems to give us a hint.

| tenet.htb/index.php/2020/12/17/logs/  This is under the tab called “migration” |
| --- |

So we are looking for a file called “sator.php” and there is a potential backup of it too. I am going to try this since it is the only lead I have at the moment. All other posts on the site are not so useful.

Going back to the initial apache site, we find sator.php is an extension.

| Sator |
| --- |

The comment we got sator from also mentioned a backup file. A common backup file extension is “.bak.” Attempting this ends up working and we download the sator.php backup

| Sator.php.bak |
| --- |

**PHP**

After getting hold of sator.php.bak, we have the following php code

| Sator.php.back |
| --- |

The above code looks interesting, especially the portion at the bottom with “unserialize.” After performing some research, it is possible to exploit this code, yet I think I am going to have a lot of trial and error with it. I will link some articles concerning this.

<https://owasp.org/www-community/vulnerabilities/PHP_Object_Injection>

<https://medium.com/swlh/exploiting-php-deserialization-56d71f03282a>

<https://riptutorial.com/php/example/14674/security-issues-with-unserialize>

The exploit in the above articles is called “object injection.” Based on what we see in the above code and with the standard sator.php site, we can possibly add our own user to the system to gain access.