

**INFORME: EJECUTIVO Y TÉCNICO**

**PENTESTING EN SISTEMAS “PIVOTING-AD”**

* Fecha: 7 de noviembre de 2024
* Cliente: Reto 20 – Team Challenge
* Consultora de Ciberseguridad: The Bridge - Accelerator
* Control de Cambios

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Versión | Documento | Fecha | Cambios | Autor | revisor | visto bueno |
| 1.1 | Informe de resultados | 13/11/2024 | Informe inicial | Víctor Martínez | Ángel /Jorge | Javier Tomás |

**Índice de Contenidos**

1. Introducción ----------------------------------------------------------------------3
2. Informe Ejecutivo ---------------------------------------------------------------3

* Presentación ---------------------------------------------------------------------3
* Alcance ----------------------------------------------------------------------------4
* Resumen de Actuaciones Practicadas -----------------------------------5
* Recomendaciones generales -----------------------------------------------6
* Reflexiones finales -------------------------------------------------------------9
* Normativa aplicable y sanciones ------------------------------------------9

1. Informe Técnico: --------------------------------------------------------------11

* Presentación -------------------------------------------------------------------11
* Fase de exploración –--------------------------------------------------------12
* Fase de explotación ----------------------------------------------------------15
* Fase de persistencia ---------------------------------------------------------18
* Conclusiones -------------------------------------------------------------------21
* Recomendaciones críticas -------------------------------------------------22
* Evaluación final ---------------------------------------------------------------24

1. Bibliografía ---------------------------------------------------------------------25
2. **INTRODUCCIÓN**

El presente informe está formado por 2 partes: un **informe ejecutivo**, menos técnico y dirigido a cargos responsables en la toma de decisiones o ejecutivos de la compañía, y un **informe técnico**, dirigido a los analistas de ciberseguridad y programadores que tengan que crear y ejecutar tareas para mitigar las vulnerabilidades explotadas, así como funciones de detección y respuesta ante amenazas, **con la finalidad** de mejorar los manuales de estrategia de la compañía en la **detección, contención y respuesta ante incidentes críticos en su sistema.**

**2. INFORME EJECUTIVO**



1. PRESENTACIÓN. – Este informe tiene como **objetivo** mostrar los resultados de las **vulnerabilidades detectadas y explotadas** en el sistema formado por 3 equipos pertenecientes a la red *“pivoting-ad”*, de acuerdo con el contrato firmado entre ambas partes, en el que permiten la explotación del sistema con la finalidad de conseguir la **autenticación y elevación de privilegios por atacantes externos,** consiguiendo **ser usuario con** **privilegios root, logrando,** ademásde **movimientos laterales entre los equipos y persistencia** en el sistema explotado.

**-** La red objetivo, consta de un equipo con S.O **Linux conectada** a **2** ordenadores con S.O. **Windows** “server 2019” en ***“Active Directory”***, necesitando para acceder a ellos credenciales que no aportan, habiendo usado para su explotación diversas herramientas de ciberseguridad, destacando alguno de sus resultados:

* Mediante herramientas de ***escaneo*** de vulnerabilidades, se han encontrado abiertos el exterior, los***puertos 21, 22 y 80****,* los cuales, se corresponden con los ***servicios FTP***(File Transfer Protocol), usado para la transferencia de archivos, la SSH (Secure Shell), utilizado para la interconexión entre los dispositivos de su sistema, de manera remota y segura, y las conexiones HTTP de los servicio web de su empresa, respectivamente.
* En el servicio FTP, se ha detectado una grave vulnerabilidad, permitiendo la**autenticación** mediante **credenciales *“anonymous”: “230”,*** a la **vista** de **cualquier atacante**,pudiendo acceder a la información sensible que pudiera encontrase en sus servidores , ademas de poder ser una posible vía de entrada no autorizada a sus sistemas.
* Ademas, se ha podido observar **sin autenticarse** en el mismo que existe un archivo llamado ***“CALL.html”***.
* Mediante herramientas de escaneo de servidores web para descubrir **directorios, archivos, subdominios** y otros puntos de entrada ocultos y menos evidentes, se ha **escasa información** para permitir conocer la estructura web de su organización, pero si la necesaria para poder acceder de forma no autorizada a su sistema mediante la ejecución de código malicioso, pudiendo exponer información sensible y representar un **riesgo de seguridad significativo** ( usuarios del sistema, etc)

**-** En resumen, se han conseguido explotar varias vulnerabilidades graves, debido a una **falta de actualización** en las tecnologías o aplicaciones, así como **accesos a zonas sensibles**, que no deberían estar abiertas al público que iremos desarrollando en el próximo punto.



1. ALCANCE. – Se ha centrado en **identificar y evaluar** las **debilidades de seguridad en el sistema**, para lograr las finalidades expuestas en el contrato, explotando algunas de las vulnerabilidades encontradas, que pueden causar daños el sistema, así como comprometer la integridad, confidencialidad y disponibilidad de los datos del mismo, **destacando**:

* Se ha podido acceder al servicio **FTP*,*** el cual presenta una grave vulnerabilidad al tener habilitado la autenticación mediante usuario ***Anonymous y contraseña 230.*** Una vez dentro del servicio, observamos que tenemos permisos de lectura y escritura en los archivos, ***permitiendo inyectar codigo*** ***malicioso*** para conseguir un acceso no autorizado vía “CLI” en la primera maquina “Linux-User”, con un ***usuario con bajos privilegios*.**
* Una vez dentro de la máquina Linux, se precede a investigar por diferentes directorios, encontrando información valiosa, que nos ha permitido encontrar las credenciales de otro usuario con mas privilegios en el sistema **“Shrek”: “onion”,** habiendo tenido que usar técnicas de desencriptado sencillo para ello
* Además, junto a las mencionadas credenciales, se ha hallado lo que **parecen** unas **credenciales de Windows**, debido a la barra invertida usada **“\”** y la estructura de la misma, pudiendo ser de una de las máquinas Windows a explotar: **EXAMPLE\testing:2021!Query.** En caso que esto fuera cierto, estaríamos ante una grave vulnerabilidad, por tener **credenciales** de una maquina de su red en **texto plano**.
* Siguiendo con el análisis, se ha detectado otra **vulnerabilidad crítica significativa**, ya que, el sistema permite obtener **permisos root al ejecutar el programa python 3.5**, sin necesitar contraseña, por lo que se procede a i**nyectar código malicioso en python,** permitiendo la conexión en modo **root** y máximos privilegios en la maquina “Linux-User”.
* Finalmente, las credenciales encontradas de Windows nos permiten acceder a la máquina **“Windows-User”** con el **usuario testing.**
* Mediante aplicaciones especificas para obtención de información para ataques en sistemas Kerberos, se obtiene un **servicio principal** **“iis\_service - (HTTP/WINDOWS)”**, obteniendo su hash, siendo “**hacheado”** mediante aplicaciones para ello, obteniendo descifrar la contraseña en plano: **LaRosalia2021**.
* Una vez obtenidas el usuario y la contraseña del servicio en un entorno de Kerberos, y con el uso de otras herramientas, se han obtenido los **hashes** de la bases de de datos local del entorno “active Directory” **(SAM[[1]](#footnote-0))** , consiguiendo los **hashes NTLM[[2]](#footnote-1)** de los usuarios del dominio y aprovechando que el sistema Kerberos permite autenticarse con el mismo, se realiza un **“ataque Pass the hash”** con la contraseña del Administrator con resultado positivo, consiguiendo una conexión vía CLI a la maquina **Windows-DC con máximos privilegios** en el sistema.



1. RESUMEN DE ACTUACIONES PRACTICADAS. – Se han realizado numerosas actuaciones, explotando ciertas debilidades / vulnerabilidades detectadas, algunas de las cuales han sido comentadas anteriormente, **consiguiendo** finalmente el **objeto del contrato**, es decir, la **autenticación** con usuario **con privilegios máximos** en la máquina **Windows-DC**, mediante la realización de **movimientos laterales** sobre las otras máquinas de la **red Pivoting-AD**, aportando detalles más técnicos más adelante.



1. RECOMENDACIONES GENERALES.- En el **análisis y explotación** reciente de seguridad de su **red Pivoting-AD**, perteneciente a la infraestructura de su organización, se han detectado **varias vulnerabilidades críticas** que requieren su atención para proteger los datos y garantizar el funcionamiento seguro de los sistemas. A continuación, se presenta un **resumen de las debilidades identificadas y las recomendaciones** para subsanarlas, mitigando los riesgos identificados y garantizando así la integridad, confidencialidad y disponibilidad de los datos y servicios, en un lenguaje accesible para facilitar su comprensión.

Los detalles técnicos de estas vulnerabilidades se explicarán, más adelante en el informe técnico correspondiente.

1. **DETECCION DE PROBLEMAS:**

* **Puertos abiertos** al exterior detectados:
* Puerto 21 (FTP) - Permite la transferencia de archivos.
* Puerto 22 (SSH) - Permite conexiones remotas seguras.
* Puerto 80 (HTTP) - Habilita el tráfico web no cifrado.

### **- Análisis de Vulnerabilidades:**

* **FTP** (Puerto 21): Este puerto presenta un riesgo alto debido a la **autenticación** configurada como **"anonymous:230"**, lo que permite **acceso sin credenciales** específicas. Esta configuración facilita la posibilidad de acceso no autorizado y modificación de archivos sensibles, abriendo una **vía potencial para intrusiones** en el sistema.
* **SSH** (Puerto 22): El servicio SSH es crítico para conexiones seguras, pero una configuración incorrecta o una **versión desactualizada** puede exponer el sistema a ataques de fuerza bruta o explotación de vulnerabilidades conocidas.
* **HTTP** (Puerto 80): El **tráfico web sin cifrar** puede **permitir** a un atacante **interceptar comunicaciones** o manipular contenido entre el cliente y el servidor.

**- Recomendaciones para Mitigación:**

* Cerrar el puerto FTP o, **preferiblemente,** reemplazarlo por **SFTP**, que asegura la transferencia de archivos mediante el cifrado SSH.
* **Actualizar SSH** a la última versión y aplicar configuraciones de seguridad avanzadas.
* Implementar **autenticación con claves públicas** para SSH en lugar de contraseñas.
* Considerar el uso de **HTTPS (Puerto 443)** en lugar de HTTP (Puerto 80) para asegurar el tráfico con cifrado SSL/TLS, p**rotegiendo la confidencialidad e integridad de los datos** en tránsito.

1. **DETECCION DE PROBLEMAS:**

* **Directorio web abierto ("Index of")**: Se detectó que el servidor web ha permitido acceso a un **directorio crítico “/files”** a través de una página de índice, lo que expone a los usuarios o atacantes a visualizar la estructura interna del servidor.
* **Riesgo de exposición:** Esta configuración puede hacer que archivos sensibles, como configuraciones, registros de errores o respaldos, sean accesibles públicamente, así como a la **ejecución de archivos maliciosos desde la web**.

**- Análisis de Vulnerabilidades:**

* **Riesgo de enumeración de archivos:** Un atacante puede acceder y enumerar archivos y directorios expuestos a través de la página "Index of", permitiéndole **comprender mejor la estructura interna del sistema**.
* **Posibilidad de acceso a archivos sensibles:** Si existen archivos **mal configurados** o que **contienen información confidencial** (como bases de datos, credenciales, o archivos de configuración), estos podrían ser leídos o descargados, facilitando posibles **ataques de escalamiento o robo de información.**
* **Incremento de la superficie de ataque:** La visibilidad de los archivos facilita el descubrimiento de vulnerabilidades o errores de configuración en el sistema, abriendo una puerta para futuros intentos de explotación.

**- Recomendaciones para Mitigación:**

* **Deshabilitar la opción de listado de directorios[[3]](#footnote-2):** Configurar el servidor web para que **no** permita el acceso público a la página **"Index of".**
* **Restringir permisos de acceso a directorios sensibles:** Asegurarse de que **sólo el personal autorizado** tenga acceso a directorios sensibles y que estos no sean accesibles públicamente.
* **Mover archivos confidenciales fuera de la raíz pública del servidor web:** Mantener archivos de configuración, registros y respaldos fuera de la raíz web y asegurarse de que solo sean accesibles internamente o a través de conexiones seguras

1. DETECCION DE PROBLEMAS:

* **Archivos detectados con credenciales inseguras:** Se han identificado archivos en el sistema que almacenan credenciales, algunos de los cuales usan hashes débiles (MD5) y otros contienen contraseñas en texto plano.
* **Riesgo de exposición:** La presencia de credenciales en texto plano y hashes inseguros aumenta el riesgo de que un atacante pueda descifrar o acceder directamente a estas contraseñas.

**- Análisis de Vulnerabilidades:**

* **Credenciales en texto plano:** Las contraseñas almacenadas en texto plano son accesibles en caso de que un atacante obtenga acceso al sistema de archivos, permitiéndole **obtener acceso a cuentas o servicios sin necesidad de descifrado**.
* **Hashes débiles:** Si las credenciales están almacenadas con algoritmos de **hash inseguros** (como MD5 o SHA-1), son **vulnerables a ataques de fuerza bruta** o ataques de diccionario, ya que estos algoritmos son rápidos de procesar y existen bases de datos públicas de hashes precalculados.
* **Facilidad para escalar privilegios y movimientos laterales:** La presencia de credenciales vulnerables facilita que un atacante, tras obtener acceso inicial, escale privilegios en el sistema o comprometa otras cuentas, servicios u otros equipos del sistema.

- **Recomendaciones para Mitigación:**

* **Reemplazar almacenamiento de contraseñas en texto plano:** Modificar las configuraciones para que las contraseñas no se guarden en texto plano. Implementar el almacenamiento de contraseñas cifradas con un estándar seguro.
* **Usar algoritmos de hashing seguros:** Reemplazar los hashes débiles por **algoritmos robustos** como bcrypt[[4]](#footnote-3), Argon2[[5]](#footnote-4), o PBKDF2[[6]](#footnote-5), que incorporan mecanismos de "salting" y son resistentes a ataques de fuerza bruta.
* **Restringir el acceso a archivos de credenciales:** Limitar los permisos de acceso para que solo usuarios o procesos autorizados puedan ver estos archivos, minimizando el riesgo de exposición.

1. DETECCION DE PROBLEMAS:

* **Permiso “NOPASSWD” en Python3.5**: Se detectó que el usuario vulnerado **“Shrek”** tiene permisos sudo configurados con la opción **NOPASSWD** para ejecutar **/usr/bin/python3.5** como root, sin requerir contraseña.
* **Riesgo de abuso**: Esta configuración permite ejecutar **comandos arbitrarios con privilegios elevados**, ya que Python puede ser utilizado para acceder al sistema de archivos, ejecutar comandos y inyectar código malicioso en el sistema.

**- Análisis de Vulnerabilidades:**

* **Ejecución de comandos como root**: Dado que Python permite ejecutar código arbitrario, un atacante que obtenga acceso a este usuario puede utilizar **sudo /usr/bin/python3.5** para ejecutar **comandos o scripts** con permisos de **root**.
* **Riesgo de escalamiento de privilegios**: Esta configuración facilita el escalamiento de privilegios, ya que el atacante no necesita autenticarse para ejecutar Python con permisos elevados. Esto permite acceder a archivos de sistema, modificar configuraciones, y comprometer otros usuarios.
* **Impacto crítico en la seguridad**: La capacidad de ejecutar Python como root sin contraseña **compromete completamente la seguridad del sistema**, dado que Python tiene acceso ilimitado al entorno y recursos del sistema.

**- Recomendaciones para Mitigación:**

* **Eliminar el permiso NOPASSWD en /usr/bin/python3.5**: Revocar el acceso a sudo sin contraseña para Python. Esto evitará que cualquier usuario ejecute Python como root sin autenticación.
* **Restringir el acceso a Python con privilegios elevados**: Si es necesario que ciertos scripts se ejecuten como root, configurar sudo para que solo el script necesario tenga permisos y no el intérprete de Python completo.
* **Implementar políticas de seguridad de sudo estrictas**: Revisar y minimizar los comandos permitidos con sudo, especialmente con la opción NOPASSWD, para reducir el riesgo de abuso.

1. **DETECCION DE PROBLEMAS:**

* **Enumeración de SPNs en el Dominio Kerberos:** Utilizando el script **PowerView.ps1**, se realizó una enumeración de los **Service Principal Names (SPN)** del controlador de dominio del entorno Active Directory, lo que **permitió identificar cuentas de servicio** configuradas para Kerberos. Estas cuentas pueden ser un objetivo, ya que suelen estar asociadas con servicios y podrían tener permisos elevados o credenciales útiles.
* **Credenciales obtenidas:** Se descifró un hash de SPN del servicio **iis\_service (HTTP/WINDOWS)**, logrando **acceder** a la cuenta **con privilegios** de servicio web.

### **- Análisis de Vulnerabilidades:**

* **Acceso mediante SPN:** La existencia de **SPNs** configurados para Kerberos expone cuentas de servicio que pueden ser **vulnerables a ataques de fuerza bruta** o de descifrado de hash, como el **ataque Kerberoasting**.
* **Privilegios del servicio web:** La cuenta del servicio **“iis\_service”** otorgó acceso a la red interna y a recursos adicionales que, con credenciales válidas, pudieron ser aprovechados para avanzar en el ataque.

- **Recomendaciones para Mitigación:**

* **Limitar el número de cuentas de servicio con SPNs:** Solo las cuentas estrictamente necesarias deben tener configurados SPNs. Eliminar SPNs innecesarios **reduce la superficie de ataque.**
* **Rotar y fortalecer contraseñas de cuentas de servicio:** Usar contraseñas **robustas y configurar una política de rotación** regular de contraseñas para cuentas de servicio.
* **Auditar accesos y configuraciones de Kerberos:** Revisar y auditar frecuentemente la configuración de Kerberos para detectar SPNs expuestos y cuentas vulnerables.

1. **DETECCION DE PROBLEMAS:**

* **Acceso a la SAM y NTLM Hashes:** A través del acceso a la cuenta de servicio obtenida, se **empleó** la colección de herramientas **Impacket[[7]](#footnote-6)** para extraer los hashes NTLM de la SAM (Security Account Manager) del sistema.

Esta acción **permitió acceder a hashes** de otras cuentas, incluidas cuentas de **privilegio más alto.**

### **- Análisis de Vulnerabilidades:**

* **Riesgo de acceso no autorizado a Hashes NTLM en la SAM:** La extracción de hashes permite que un atacante realice ataques de Pass-the-Hash o intente descifrar estos hashes para obtener credenciales en texto claro.
* **Escalabilidad del ataque:** La posibilidad de extraer estos hashes incrementa la probabilidad de un ataque de **escalamiento de privilegios o de movimiento lateral en la red.**

- **Recomendaciones para Mitigación:**

* **Asegurar el Archivo SAM y Hashes NTLM:** Limitar el acceso a la SAM y evitar que cuentas de servicio puedan acceder a los hashes NTLM almacenados, i**mplementando **políticas de restricción de acceso mas robustas.****
* **Deshabilitar LM y NTLMv1 en el dominio**: Configurar la seguridad de la red para deshabilitar los hashes LM y NTLMv1, lo cual evita que se almacenen hashes inseguros en la SAM.
* **Implementar protección contra ataques de Pass-the-Hash**: Usar tecnologías como **Credential Guard**[[8]](#footnote-7) en entornos Windows para proteger las credenciales y mitigar el riesgo de explotación de Pass-the-Hash.

1. **DETECCION DE PROBLEMAS:**

* **Pass-the-Hash para Escalamiento de Privilegios:** Utilizando los hashes NTLM obtenidos, se realizó unPass-the-Hash en otra máquina Windows, **logrando acceso con máximos privilegios**, lo que facilitó el **control total sobre el sistema**.

### **- Análisis de Vulnerabilidades:**

* **Pass-the-Hash para Escalamiento de Privilegios:** La vulnerabilidad de Pass-the-Hash permite a un atacante **autenticarse con privilegios elevados** **utilizando un hash NTLM en lugar de la contraseña**, lo que facilita el acceso total sin necesidad de autenticación adicional.
* **Compromiso total del sistema:** Este método otorga al atacante control completo sobre la máquina de destino, comprometiendo así toda su seguridad y acceso a datos críticos.

- **Recomendaciones para Mitigación:**

* **Implementar autenticación multifactor (MFA):** Agregar un segundo factor de autenticación puede evitar que el uso de hashes NTLM sea suficiente para acceder a sistemas críticos.
* **Segregar cuentas de administrador y usuarios estándar:** Minimizar el uso de cuentas con privilegios administrativos y usar cuentas con privilegios limitados para tareas rutinarias.
* **Auditar y monitorear el uso de credenciales de alto privilegio:** Implementar un sistema de auditoría y monitorización para identificar intentos de Pass-the-Hash y alertar sobre posibles actividades sospechosas.

Además de las acciones mencionadas, y con carácter general, se recomienda evaluar y actualizar la política de seguridad de la empresa hacia el **modelo de seguridad “Zero Trust”**[[9]](#footnote-8). el cual, fortalecerá significativamente la postura de seguridad de la empresa al reducir la superficie de ataque y garantizar que sólo los usuarios autorizados puedan acceder a los datos críticos.



1. REFLEXIONES FINALES

Si bien, algunas de estas recomendaciones requieren un enfoque más técnico, es vital entender la importancia de la implementación de estas recomendaciones, las cuales, reducirán considerablemente las posibilidades de un ataque exitoso y mejorará la seguridad general de la infraestructura organizacional, evitando riesgos graves y potenciales violaciones de seguridad, sugiriendo que los equipos técnicos, desarrolladores y de seguridad trabajen de manera conjunta para implementar estas soluciones a la mayor brevedad posible.

El informe técnico detallado proporcionará un análisis más profundo y pasos específicos para abordar cada vulnerabilidad.

1. NORMATIVA APLICABLE Y SANCIONES

Existen diversas normativas que regulan la protección de datos y la seguridad de la información, y que podrían ser aplicables en este caso:

* 1- Reglamento General de Protección de Datos (RGPD)[[10]](#footnote-9) y la Ley Orgánica de Protección de Datos Personales y garantía de los derechos digitales (LOPDGDD)[[11]](#footnote-10). - Si la información confidencial que se encuentra en el sistema no se encuentra debidamente custodiada, su incumplimiento podría acarrear sanciones importantes para la empresa.
* 2- Ley de Servicios de la Sociedad de la Información y el Comercio Electrónico (LSSI)[[12]](#footnote-11). - Los prestadores de servicios (corporaciones, empresas, etc) deben adoptar las medidas técnicas y organizativas necesarias para garantizar la seguridad de los datos de los usuarios, pudiendo su incumplimiento acarrear sanciones para la empresa.
* 3- Directiva NIS2[[13]](#footnote-12). - En caso de comprometer a infraestructuras críticas o servicios esenciales, las empresas pueden enfrentarse a sanciones administrativas y reputacionales por no cumplir con los estándares mínimos de ciberseguridad exigidos.
* 4- ISO - 27001[[14]](#footnote-13).- Estándar internacional que ayuda a las empresas a identificar, gestionar y mitigar riesgos de ciberseguridad, estableciendo los requisitos de un SGSI[[15]](#footnote-14), el cual proporciona el marco de protección para la triada CIA, asegurando que la organización cumple con los requisitos legales y normativos vigentes, y protege eficazmente sus datos contra amenazas, como el acceso no autorizado, la pérdida o la corrupción de la información, facilitando, paralelamente, el cumplimiento de la directiva NIS2.
* 4 - NIST - CIBERSECURITY FRAMEWORK[[16]](#footnote-15).- Proporciona una estructura integral a las organizaciones, con la finalidad de evaluar y mejorar la seguridad de los sistemas de información, desde una perspectiva que permite a las organizaciones personalizar sus estrategias de ciberseguridad según sus necesidades.

Estas estrategias, aseguran la protección de sus activos críticos, la detección temprana de amenazas, y una respuesta rápida ante incidentes de manera efectiva.

Las sanciones por el incumplimiento de las normativas de protección de datos y seguridad de la información pueden ser de elevado valor, por ejemplo, en el caso del RGPD, las multas pueden ascender hasta el 4% del volumen de negocio mundial anual de la empresa o 20 millones de euros, lo que sea mayor y en el caso de la LOPDGDD, las multas pueden ascender hasta 300.000 euros. -Además, la empresa está obligada a notificar a las autoridades y a los afectados en un plazo determinado las consecuencias del incidente, pudiendo agravar la repercusión pública del incidente a la reputación de la empresa.

**3.- INFORME TÉCNICO**

1. PRESENTACIÓN. – Para conseguir el objetivo fijado en el contrato, se han seguido la siguiente línea de investigación:

* El Equipo ha sido entregado con un sistema **Linux 3.11.0-15**-generic i686*,* sin aportar credenciales de inicio de sesión, por lo que, el análisis y explotación será realizado sin acceso a información interna de la organización.
* Para esta explotación se ha usado como maquina **atacante**, un **host** de **Kali Linux**, en su **versión .3 2024**, conectando en modo “*bridge”* con la máquina virtualizada objeto del presente.
* INFORMACIÓN INICIAL. - Una vez conocida la IP asignada a sistema objetivo, se procede a consultar mediante Nmap*,* herramienta de código abierto utilizada para explorar y auditar la seguridad de redes y sistemas*,* el rango de IPs donde se encuentran ambas maquinas*,* siendo la de Bside\_Vancouver\_2018 (BV18, en adelante): 192.168.1.243 y de la maquina atacante: 192.168.1.134. Además, la maquina objetivo tiene un total de 3 puertos abiertos:

|  |  |  |  |
| --- | --- | --- | --- |
| **Puerto** | **21** | **22** | **80** |
| **Servicio** | FTP | SSH | HTTP |
| **Versión** | VSFTPD 2.3.5 | OpenSSH 5.9p1 -Debian 5Ubuntu 1.10 | Apache 2.2.22(Ubuntu) |

* Además, se confirma la presencia del directorio ***/robots.txt***, siendo éste, un archivo que los administradores de sitios web colocan en la raíz de su servidor, para dar instrucciones a los motores de búsqueda sobre cómo rastrear e indexar las páginas de la web, permitiendo si es visible, aportar información de la estructura de la web, así como informaciones sensibles. Dentro de este archivo se ha encontrado un directorio que ha sido clave para la explotación de esta máquina: **“/backup\_wordpress”.**
* En la línea de investigación seguida se han explotado, en primer lugar, varias vulnerabilidades localizadas a través del puerto 80 y sus directorios web hasta conseguir acceso a una reverse shell de acceso al sistema, donde se han explotado otras vulnerabilidades encontradas en la configuración del Kernel de la maquina objetivo, como iremos explicando con mas detalle más abajo.

1. FASE DE EXPLORACIÓN

A.- **USO APLICACIÓN GOBUSTER:**

* Herramienta de seguridad y hacking web, comúnmente utilizada durante las fases de reconocimiento en pruebas de penetración, que usa *“fuerza bruta”* para descubrir objetos y directorios ocultos o no indexados en un servidor web, habiendo sido usada en el puerto 80, pero en con múltiples directorios que iban apareciendo en el análisis:
* En el escaneo a la IP únicamente se han encontrado 2 directorios accesibles, uno, que ya conocíamos (robots.txt y robots) y otro directorio /index, ***no arrojando datos relevantes*** para esta explotación.

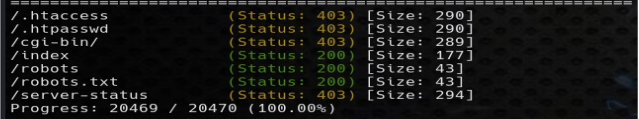


Imagen 1[[17]](#footnote-16)

* En un segundo escaneo, usando la ***IP junto*** ***al directorio*** que nos da acceso a la web de wordpress del sistema atacado: ***/backup\_wordpress***, se consiguen accesos a varios subdominios, destacando para esta explotación: ***/wp-login.***

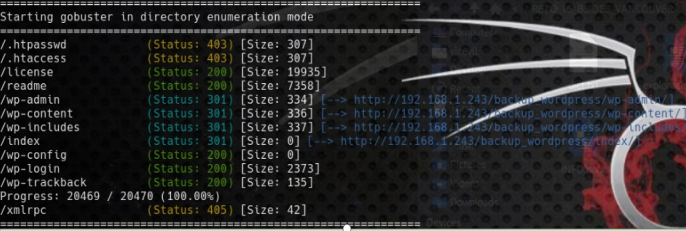


Imagen 2[[18]](#footnote-17)

* Se ha seguido haciendo ***“fuzzing[[19]](#footnote-18)”*** de cada uno de los directorios, pero aunque parecen accesibles (codigo 200), al final no aportan información o dan como resultado un código 404, pero si se ha ***conseguido conocer la infraestructura interna del servidor web***, incluso con directorios llamados “***users”*** , por lo que es recomendable revisar la configuración de la web para que no haya tantos directorios accesibles a herramientas fuzzing.



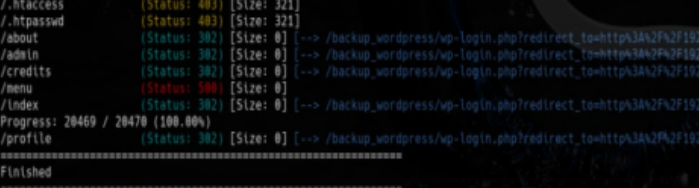


Imagen 3[[20]](#footnote-19)

- En definitiva, se han consultado todos los dominios y subdominios, extrayendo una información general de la estructura del servidor, pero ningún dato concreto para la explotación.

B.- **EXPLORACION WEB -** [http://192.168.1.243/backup\_wordpress/](http://192.168.1.243/backup_wordpress/:)

- Se ha analizado el contenido de la web, sobre los diferentes cuadros de textos de comentarios, sobre la misma url, los enlaces, los archivos de descarga y sobre la sección de login con usuario y contraseña, destacando:

- Al hacer ‘click’ en la ***sección “META”***, en el enlace “***Entries RSS”***, descarga automáticamente un archivo de nombre “28APAM4W”,el cual, contiene ***código HTML,*** que si se lee detenidamente, se puede observar que nombra a un ***administrador IT llamado “john”***

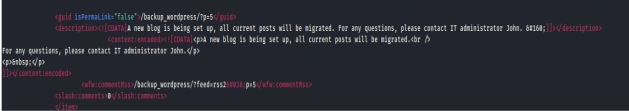


Imagen 4[[21]](#footnote-20)

- En la ***página de login***, hay un enlace para ***restablecimiento de la contraseña*** . Si pulsamos en ese enlace, nos dirige a una sección de la web, donde solicita ***introducir “username o email”***, probando con ***admin y john***,siendo usuarios validos, ya que te dirige a una web informado que no han podido enviar el email, y con otros usuarios probados sale inmediantamente un error.

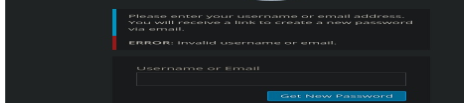


Imagen 5[[22]](#footnote-21)

- Se localiza en una “***página index of[[23]](#footnote-22)”*** <http://192.168.1.243/backup_wordpress/wp-admin/maint> , la cual es, un directorio del servidor web que ***lista automáticamente los archivos y carpetas disponibles*** en esa ubicación, permitiendo a los usuarios navegar, leer y en muchos casos descargar archivos (en nuestro caso no), pudiendo exponer información sensible, ayudando en los ataques malintencionados, ***representando un riesgo de seguridad significativo***.



Imagen 6[[24]](#footnote-23)

- Al realizar ‘click’ sobre el ***archivo rapair.php***, nos redirige a una página donde informa lo que debes de hacer en caso de problemas con la base de datos, aportando el ***directorio “wp- config”***,una línea de código y un ***enlace con las 8 claves maestras del servidor,*** las cuales, son generadas aleatoriamente y se usan para mejorar la seguridad de WordPress en su autenticación, en tareas como: cifrado de las cookies de autenticación de los usuarios, asegurando que las sesiones no puedan ser falsificadas y en la protección de los tokens de autenticación. Aunque directamente no se pueden extraer las contraseñas de los usuarios de estas claves, si ***existe un riesgo crítico al sistema*** poder acceder a ellas, ya que, usuarios no autorizados podrían realizar *acciones maliciosas* sobre el sistema como: ***secuestro de sesiones***[[25]](#footnote-24), acceso total al sistema[[26]](#footnote-25), desactivar protecciones de seguridad[[27]](#footnote-26), instalación de “***backdoors***[[28]](#footnote-27)” y robo de información.

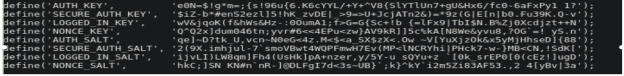


Imagen 7[[29]](#footnote-28)

1. FASE DE EXPLOTACIÓN:

- Mediante el uso de la herramienta ***Hydra***[[30]](#footnote-29), junto con los usuarios válidos conseguidos en la fase anterior, se ha efectuado un ataque mediante diccionario con resultado positivo, obteniendo las ***credenciales: john:enigma***, con las que tenemos ***acceso con privilegios de administrador*** a la ***web “Deprecated wordpress blog”.***

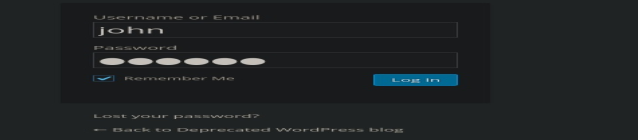


Imagen 8[[31]](#footnote-30)

- Una vez ***‘logueado’***, se procede a realizar una inspección sobre las diferentes opciones que ofrece la página, con la finalidad de buscar alguna ***vulnerabilidad*** que permita obtener una conexión remota al sistema, encontrando una en la ***sección de “Plugins”***, ya que, ***permite modificar e instalar*** plugins predeterminados, pero también personalizados, subidos desde la red local, apareciendo junto a los plugins la ***opción “Activate”***



Imagen 9[[32]](#footnote-31)

- Se consulta en la web como debe ser la estructura de los plugins de wordpress, realizando un ***archivo*** con el ***encabezado exigido*** por la aplicación **más una reverse shell**, siendo el mismo comprimido en .zip[[33]](#footnote-32), con la finalidad que lo reconozca wordpress como plugin válido, teniendo como ***resultado el acceso*** a una shell con un ***usuario*** ***con escasos privilegios (ww-data).***

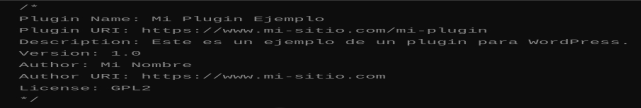


Imagen 10[[34]](#footnote-33)

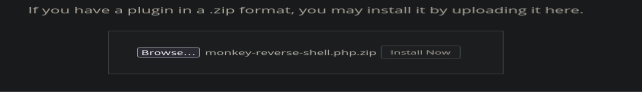


Imagen 11[[35]](#footnote-34)

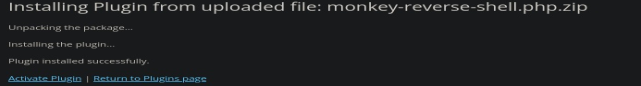


Imagen 12[[36]](#footnote-35)

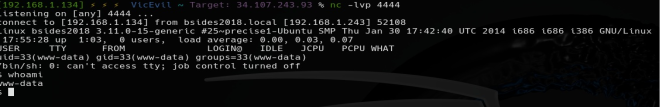


Imagen 13[[37]](#footnote-36)

- Una vez dentro del sistema, se realiza una búsqueda de posibles vulnerabilidades en la configuración de su Kernel (archivos SUID[[38]](#footnote-37), consulta en la web exploits específicos para tecnologías instaladas, GTOFbins[[39]](#footnote-38), etc), encontrando finalmente una ***vulnerabilidad*** en un ***archivo del cron***, concretamente en el directorio ***“/usr/local/bin/cleanup”*** , el cual, tiene **permisos de usuario de root**, pero permite la lectura y escritura por parte del usuario ww-data, siendo copiado al directorio “***/tmp”*** donde nuestro usuario tiene plenos permisos para su modificación.

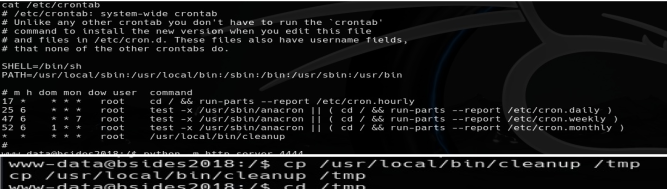


Imagen 14[[40]](#footnote-39)

- Prosiguiendo con la vulnerabilidad encontrada, se realiza un ***script*** en ***“bash[[41]](#footnote-40)”*** con una ***reverse shell en python***, denominándolo ***“cleanup”,*** sustituyendo nuestro archivo malicioso, directamente en el directorio donde se esta ejecutando el archivo original: “***/usr/local/bin/”****,* no sin antes haberle dado permisos de ejecución.

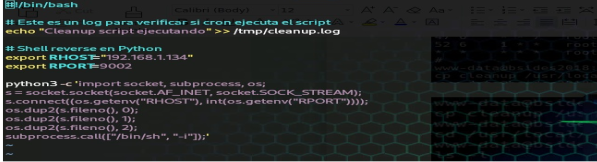


Imagen 15[[42]](#footnote-41)

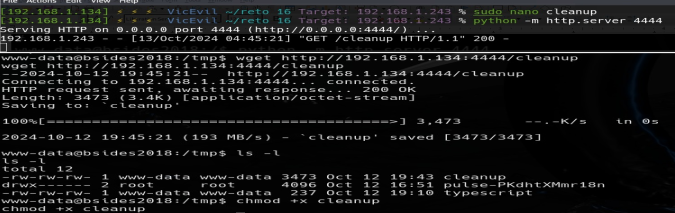


Imagen 16[[43]](#footnote-42)

- Una vez esperado el tiempo indicado (1 minuto), se consigue acceso mediante la reverse shell con máximos privilegios en el sistema “root”, habiendo finalizado la fase de escalada de privilegios.

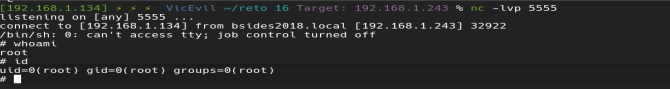


Imagen 17[[44]](#footnote-43)

- Como usuario “root” del sistema, se realiza un “cat” al archivo “shadow”, lugar donde se encuentran todas las contraseñas cifradas de los usuarios del sistema, con la finalidad de intentar descifrar las que aparecen en la imagen de abajo.



Imagen 18[[45]](#footnote-44)

- Mediante el uso de la herramienta ***hashcat***, utilizada principalmente para crackear (descifrar) contraseñas que han sido cifradas en forma de hashes, usando su ***modalidad de “ataque mediante diccionario” y el atributo “--quiet”*** (no muestra la salida en pantalla y solo los resultados finales positivos), se consigue descifrar, únicamente, la contraseña del usuario ***“anne”***, siendo ésta ***“princess”.***



Imagen 19[[46]](#footnote-45)

1. FASE DE PERSISTENCIA.

* Una vez con máximos privilegios en el sistema, se procede a realizar **persistencia** en el mismo, usando ***un script*** con una reverse shell , ***junto*** a la creación de ***un servicio*** que se ejecute en el ***directorio /etc/init.d[[47]](#footnote-46)*** y llame a al script cada vez que un usuario inicie el sistema.
* En primer lugar utilizaremos un script con una reverse-shell que sera enviada a la maquina objetivo, mediante el método “servidor python - wget”, a la ubicación /usr/local/bin/php-reverse-shell.php.

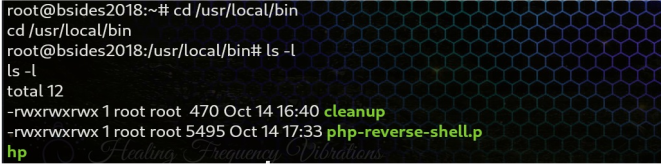


Imagen 20[[48]](#footnote-47)

- En segundo lugar, se prepara el ***archivo de creación del servicio*** en bash, el cual, ejecutará la reverse shell cada vez que cualquier usuario inicie el sistema. Éste comienza con su ***encabezado (LSB)***[[49]](#footnote-48), que proporcionará al sistema información sobre como manejar el servicio denominado ***“reto\_16”*** durante su ejecución, no siendo obligatorios pero si recomendados para evitar anomalías en la ejecución del mismo. Después, viene el código de ejecución, nombrando una variable ***con la ruta de ejecución de la reverse shell,*** comenzando el condicional, habiendo realizado únicamente la parte del servicio que inicia la ejecución, que es lo que nos interesa, dejando a un lado stop, restart, etc.

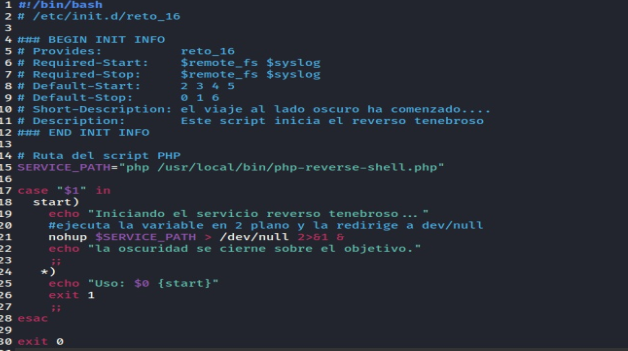


Imagen 21[[50]](#footnote-49)

- En este punto, para que se inicie con el sistema es necesario agregarla a las aplicaciones y scripts que se inician con el sistema independientemente del usuario que lo haga, para ello, se ejecuta el comando: ***update-rc.d reto\_16 defaults.***

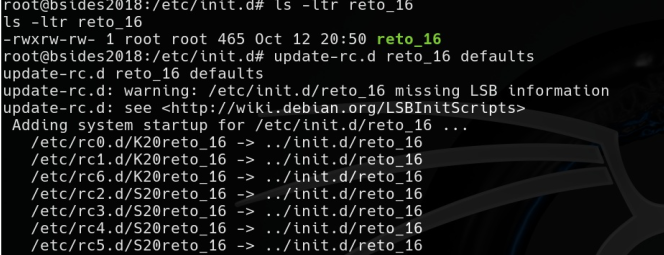
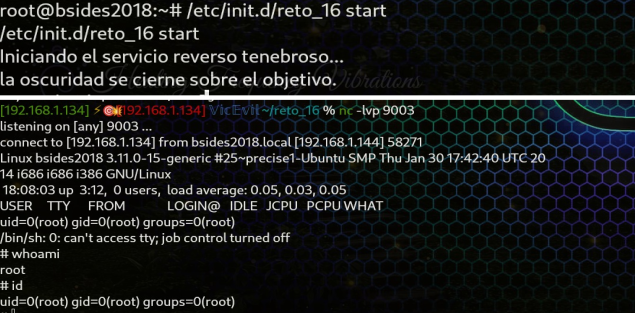


Imagen 22[[51]](#footnote-50)

- Finalmente, probamos el servicio, ejecutando el archivo ***“reto\_16”*** en la maquina objetivo, devolviendo una reverse shell en la Kali con privilegios root, el cual se ejecutara cada vez que se reinicie la maquina.



1. CONCLUSIONES

- El análisis y explotación realizados en el sistema auditado BC18, han revelado varias vulnerabilidades críticas que podrían ser explotadas por actores maliciosos para obtener acceso no autorizado, elevación de privilegios y persistencia en el sistema, las cuales, ponen en riesgo la confidencialidad, integridad y disponibilidad de su organización, destacando:

- A lo largo de la investigación, se han identificado servicios expuestos al exterior, como FTP, SSH y HTTP, con configuraciones inseguras y versiones desactualizadas que representan un alto riesgo de explotación (autenticación anónima en FTP y visibilidad archivo robots.txt, acceso a directorios sensibles: /backup\_wordpress, etc), los cuales, han permitido obtener información clave sobre la estructura del servidor y usuarios válidos del sistema.

- Mediante técnicas de enumeración de directorios y análisis de archivos públicos, se obtuvo acceso a archivos de configuración críticos como "Index of", que incluía las claves maestras de WordPress o authentication keys, que podría haber sido aprovechado para comprometer, aún más, la autenticación de usuarios y la integridad del sistema.

- El uso de herramientas de fuerza bruta como Hydra, ha permitido obtener credenciales válidas, lo que facilitó el acceso a la interfaz administrativa de WordPress. A través de este acceso, se ha logrado subir y ejecutar un plugin malicioso que otorgó acceso remoto al sistema, aunque con permisos limitados en un principio. Sin embargo, la explotación de un archivo mal configurado con permisos máximos (root) en crontab y su manipulación mediante un script malicioso, ha permitido escalar privilegios, obteniendo finalmente control total sobre el sistema.

- Posteriormente se han explotado los hashes de contraseñas del directorio: /etc/shadow, a través de la herramienta hashcat, permitiendo descifrar las credenciales de un usuario adicional, reforzando aún más el control sobre el sistema comprometido.

- Finalmente, para consolidar el acceso permanente , se implementó un mecanismo de persistencia mediante la creación de un servicio en init.d, garantizando la conexión remota continua incluso tras reinicios del sistema.

En conclusión, las debilidades encontradas, muchas de ellas explotadas con éxito, ponen de manifiesto ***l*a necesidad urgente de revisar las configuraciones de seguridad, actualizar las aplicaciones críticas y reforzar los mecanismos de autenticación y control** de acceso en la infraestructura evaluada.

1. RECOMENDACIONES CRÍTICAS:

A continuación, se presentan las recomendaciones más urgentes para subsanar las vulnerabilidades detectadas y mejorar significativamente la seguridad de la infraestructura:

1. **Cerrar o asegurar los puertos** abiertos (FTP, SSH, HTTP):

* **FTP:** Eliminar la opción de autenticación anónima o, preferiblemente, reemplazar FTP por el ***servicio SFTP***, el cual incluye autenticación mediante SSH, protegiendo la transferencia de archivos. En caso que n***o*** sea necesario este puerto, ***cerrar el mismo*** es la mejor solución.
* **SSH:** Actualizar OpenSSH a su versión más reciente (v9.9) y ***limitar el acceso al puerto 22*** mediante listas blancas de IP o, aún mejor, mediante autenticación basada en claves públicas, deshabilitando, en este último caso, el uso de contraseñas.
* **HTTP:** ***Migrar*** el servicio ***HTTP a HTTPS*** para cifrar las comunicaciones y evitar la exposición de información en texto plano.

1. **Actualizar el software del servidor**:

* Actualizar inmediatamente todos los servicios críticos, como el ***servidor web Apache y WordPress***, a sus versiones más recientes, corrigiendo así vulnerabilidades conocidas.
* Asegurarse de que los ***parches de seguridad*** estén aplicados de forma continua y establecer un proceso de actualización regular.

1. **Proteger directorios** sensibles **y archivos** de configuración:

* ***Restringir*** el acceso al archivo “***robots.txt”*** y evitar la exposición de directorios críticos como ***/backup\_wordpress***, entre otros.
* Configurar correctamente el servidor web para ***deshabilitar*** la opción de ***listado de directorios "Index of"*** y evitar la exposición pública de archivos y carpetas.

1. Mejorar la seguridad en el **inicio de sesión y gestión de contraseñas:**

* Implementar autenticación ***multifactor (MFA)*** para todas las cuentas administrativas, lo que añade una capa adicional de protección frente a ataques de fuerza bruta.
* Exigir ***contraseñas fuertes*** y habilitar ***políticas de cambio periódico*** de contraseñas para todos los usuarios del sistema.
* Restringir la funcionalidad de restablecimiento de contraseñas expuesta, reforzándola para que no revele información sobre usuarios válidos, aportando el ***mismo mensaje que caso de error en el login.***

1. Controlar el acceso a **tareas programadas (crontab)**:

* Asegurar que los ***archivos y scripts*** ejecutados por “crontab” tengan los ***permisos adecuados*** y sean revisados periódicamente, para evitar que puedan ser manipulados por usuarios no autorizados.

1. Aumentar la visibilidad y control del sistema mediante **herramientas de monitorización y auditoría**:

* ***Implementar*** herramientas de detección de intrusiones ***(IDS/IPS)*** y sistemas que monitoricen actividades sospechosas ***(EDR/XDR[[52]](#footnote-51))***, con la finalidad de detectar posibles accesos no autorizados en tiempo real.
* Configurar ***registros detallados de acceso*** y revisarlos regularmente para identificar posibles brechas de seguridad, ya sea, a través de ***aplicaciones SIEM*** o similares, así como, habilitando y configurando el registro de evento a través de la aplicación ***“rsyslog”****,* generalmente está instalado en la mayoría de las distribuciones modernas de Linux

1. **Reforzar la política de persistencia** y revisar servicios del sistema:

* ***Revisar*** los ***servicios*** que se ejecutan automáticamente al i***nicio del sistema*** (como aquellos en init.d) para asegurar que no existan scripts maliciosos.
* Implementar medidas de ***control de integridad sobre los archivos*** ***críticos*** del sistema para evitar la manipulación de servicios clave.

1. Revisar y gestionar **claves criptográficas** de forma segura:

* Regenerar periódicamente las claves maestras de WordPress y asegurarse de que estén protegidas adecuadamente, evitando su exposición en directorios o archivos accesibles públicamente.
* Utilizar ***gestores de claves seguros*** para almacenar y proteger todas las claves criptográficas sensibles.

En general, **aplicando estas recomendaciones** de manera inmediata, reducirá significativamente las brechas de seguridad identificadas de su organización, fortaleciendo la protección del sistema frente a ataques y accesos no autorizados, **mejorando la postura de seguridad de su empresa.**

1. EVALUACIÓN FINAL:

El análisis exhaustivo de la infraestructura auditada ha puesto de manifiesto serias deficiencias de seguridad que deben ser abordadas con carácter urgente y la peligrosidad que representa para la ciberseguridad, los accesos no autorizados a datos sensibles, pudiendo afectar a la integridad y la confidencialidad.

El sistema presenta vulnerabilidades en múltiples frentes, incluyendo la exposición de servicios no seguros, software desactualizado, y la falta de protección adecuada de archivos y directorios sensibles, lo que ha permitido su explotación, siendo fallos que representan riesgos significativos tanto para la integridad de los datos como para la disponibilidad y confidencialidad de los sistemas críticos de la organización, subrayando la criticidad de las debilidades encontradas y la facilidad con la que un atacante podría tomar el control total del sistema.

En conclusión, la **evaluación final** deja claro que el sistema, en su estado actual, es altamente vulnerable a ataques internos y externos, lo que representa un ***riesgo crítico extremo*** para la seguridad de la organización, siendo **imprescindible implementar de inmediato las recomendaciones** proporcionadas, incluidas las actualizaciones de software, la protección de servicios expuestos y la aplicación de medidas más estrictas para el control de acceso y autenticación.

Únicamente, mediante la aplicación de estas acciones, se podrá **asegurar la continuidad operativa del sistema** y minimizar las posibilidades de intrusión y explotación futura.

4.- BIBLIOGRAFÍA

<https://www.nist.gov/publications/zero-trust-architecture>

<https://commission.europa.eu/law/law-topic/data-protection/reform/rules-business-and-organisations/enforcement-and-sanctions/sanctions/what-if-my-companyorganisation-fails-comply-data-protection-rules_es>

<https://ayudaleyprotecciondatos.es/2019/02/19/sanciones-rgpd-lopd-2019/>

<https://www.nist.gov/>

<https://www.ccn.cni.es/es/normativa/directiva-nis2>

[https://gtfobins.github.io/#](https://gtfobins.github.io/)

1. Security Account Manager) es una base de datos en sistemas Windows que almacena información de cuentas de usuario, incluyendo los hashes de contraseñas, siendo crucial para autenticación en sistemas locales (con hashes LM y NTLM). [↑](#footnote-ref-0)
2. tipo de hash de contraseñas usado en sistemas Windows, basado en el algoritmo MD4 y sin utilizar un "salt", lo que hace que dos contraseñas iguales generen el mismo hash, facilitando ataques de fuerza bruta y de diccionario, siendo relativamente fácil de romper con herramientas como Hashcat o John the Ripper. [↑](#footnote-ref-1)
3. Esto se puede hacer en servidores como Apache usando la directiva Options -Indexes o en Nginx eliminando la opción de autoindex. [↑](#footnote-ref-2)
4. Función de hashing diseñada para ser computacionalmente intensiva, lo que dificulta los ataques de fuerza bruta. Además, bcrypt incorpora salting, un mecanismo que añade datos aleatorios al proceso de hashing, lo que hace que contraseñas idénticas generen hashes diferentes. [↑](#footnote-ref-3)
5. Función de hashing diseñadas igual que la anterior, pero a la que le incorporan para una mayor seguridad el uso de la memoria y numero de núcleos CPU, haciéndolo más resistente a ataques de fuerza bruta y a ataques con hardware especializado. [↑](#footnote-ref-4)
6. función de derivación de clave utilizada para proteger contraseñas que utiliza una función hash repetida muchas veces sobre la contraseña combinada con un valor salt (aleatorio) para generar un hash seguro. La cantidad de repeticiones es configurable, aumentando la resistencia contra ataques de fuerza bruta. [↑](#footnote-ref-5)
7. conjunto de herramientas y bibliotecas en Python que facilita la creación y ejecución de scripts de red para atacar y manipular protocolos de red, especialmente en entornos de Active Directory (AD), capaz de interactuar con varios protocolos de red como SMB, LDAP, y RPC, que son esenciales en entornos Windows y AD. [↑](#footnote-ref-6)
8. característica de seguridad en Windows que protege las credenciales de los usuarios (como hashes NTLM y tickets Kerberos) mediante la virtualización. [↑](#footnote-ref-7)
9. Zero Trust, parte de la premisa de no confiar en ningún usuario, dispositivo o sistema dentro o fuera de la red organizacional y se basa en los siguientes principios clave:

   - Verificación continua: La identidad y la autorización de cada usuario y dispositivo se verifican constantemente.

   - Principio de Menor privilegio: Los usuarios y dispositivos solo reciben acceso a los recursos que necesitan para realizar su trabajo.

   - Segmentación: La red se segmenta en zonas para limitar el acceso, contención de amenazas y evitar el movimiento lateral de las mismas

   - Protección de datos: Los datos se protegen con cifrado adecuado y otras medidas de seguridad.

   - Monitoreo y respuesta: La actividad de la red se monitorea constantemente para detectar y responder a las amenazas. [↑](#footnote-ref-8)
10. El RGPD es un reglamento de la Unión Europea que establece normas estrictas para la protección de datos personales [↑](#footnote-ref-9)
11. La LOPDGDD es ley española que desarrolla el RGPD y que establece normas específicas para la protección de datos personales en España [↑](#footnote-ref-10)
12. La LSSI es una legislación española que regula la prestación de servicios de la sociedad de la información y el comercio electrónico, estableciendo una serie de obligaciones a las empresas e infracciones en caso de incumplimiento., [↑](#footnote-ref-11)
13. Directiva NIS2 (Seguridad de Redes y Sistemas de Información 2) es una actualización de la Directiva NIS original, aprobada por la Unión Europea, con el objetivo de fortalecer la ciberseguridad en los sectores esenciales y en las infraestructuras críticas de los Estados miembros de la UE. [↑](#footnote-ref-12)
14. Norma internacional que define los requisitos para establecer, implementar, mantener y mejorar un sistema de gestión de seguridad de la información (SGSI), cubriendo aspectos como, el control de acceso, la gestión de incidentes de seguridad y la continuidad del negocio, siendo ampliamente utilizada para demostrar el compromiso de una organización con la ciberseguridad y la protección de datos. [↑](#footnote-ref-13)
15. Sistema de Gestión de Seguridad de la Información (SGSI), es un conjunto de políticas, procedimientos, procesos y controles implementados por una organización para gestionar, proteger y asegurar la confidencialidad, integridad y disponibilidad de la información. [↑](#footnote-ref-14)
16. Marco desarrollado por el Instituto Nacional de Estándares y Tecnología de EE. UU., diseñado para ayudar a las organizaciones a gestionar eficazmente los riesgos de ciberseguridad. Este marco se basa en cinco funciones clave: Identificar, Proteger, Detectar, Responder y Recuperar. [↑](#footnote-ref-15)
17. imagen 1.- escaner mediante gobuster de la IP atacada 192.168.1.243 [↑](#footnote-ref-16)
18. imagen 2.- escaner mediante gobuster de la IP atacada 192.168.1.243/backup\_wordpress [↑](#footnote-ref-17)
19. Técnica de prueba de seguridad que consiste en enviar grandes cantidades de datos aleatorios o malformados a un programa o sistema para identificar vulnerabilidades, errores o fallos. El objetivo es observar cómo el sistema reacciona ante entradas inesperadas, lo que puede revelar debilidades que podrían ser explotadas por atacantes. [↑](#footnote-ref-18)
20. imagen 3.- escaner mediante gobuster de la IP atacada 192.168.1.243/backup\_wordpress/wp-admin/users [↑](#footnote-ref-19)
21. imagen 4.- Parte del codigo del archivo “28APAM4W” donde se observa el usuario del administrador de la web [↑](#footnote-ref-20)
22. imagen 5.- parte de la pagina para el restablecimiento de la cuenta en caso de olvidar la contraseña. [↑](#footnote-ref-21)
23. lista automática de archivos y directorios que aparece cuando un servidor web no tiene configurado un archivo de índice predeterminado, como "index.html" o "index.php" [↑](#footnote-ref-22)
24. imagen 6.- página index of donde se observa un archivo significativo *“rapair.php”.* [↑](#footnote-ref-23)
25. Las llaves maestras se usan para cifrar las cookies de autenticación. Si alguien obtiene acceso a ellas, podría desencriptar las cookies de los usuarios, lo que permitiría suplantar su identidad y acceder a sus cuentas sin necesidad de la contraseña. [↑](#footnote-ref-24)
26. un atacante podría falsificar los tokens de autenticación y crear sesiones válidas, lo que le permitiría acceder al panel de administración de WordPress [↑](#footnote-ref-25)
27. un atacante tiene estas llaves, podría las protecciones contra la falsificación de sesiones y los ataques CSRF (Cross-Site Request Forgery)

    , facilitando la explotación de vulnerabilidades [↑](#footnote-ref-26)
28. puertas traseras) para mantener el acceso futuro incluso si se cambian las contraseñas o clave [↑](#footnote-ref-27)
29. imagen 7.- claves maestras del servidor web wordpress del sistema, también llamadas: salt keys o authentication keys [↑](#footnote-ref-28)
30. usada para realizar ataques de autenticación en varios servicios y protocolos, permitiendo probar rápidamente diferentes combinaciones de usuarios y contraseñas en múltiples servicios, como SSH, FTP, HTTP, LOGIN,entre otros. [↑](#footnote-ref-29)
31. página http://192.168.1.144/backup\_wordpress/wp-login.php donde probamos las credenciales john:enigma con resultado positivo. [↑](#footnote-ref-30)
32. página http://192.168.1.144/backup\_wordpress/wp-admin/ del administrador del servidor wordpress, donde se muestra las opciones de los plugins. [↑](#footnote-ref-31)
33. Formato que comprime archivos, lo que facilita su transporte, almacenamiento y gestión, y se puede descomprimir fácilmente para recuperar los archivos originales. [↑](#footnote-ref-32)
34. Ejemplo de encabezado o información inicial del script, que debe ir junto a la reverse shell, siendo todo comprimido en formato .zip, con la finalidad que sea reconocido por wordpress como plugin válido [↑](#footnote-ref-33)
35. Parte de la página http://192.168.1.144/backup\_wordpress/wp-admin/plugin-install.php?tab=upload, a través de la cual se suben los archivos locales, en este caso nuestro script malicioso. [↑](#footnote-ref-34)
36. Una vez procesado el envío del archivo malicioso, muestra que el mismo ha sido subido e instalado de manera satisfactoria. [↑](#footnote-ref-35)
37. Resultado de la reverse shell conseguida de acceso al sistema con el usuario “ww-data” con permisos limitados. [↑](#footnote-ref-36)
38. Set User ID del sistema, los cuales cuentan con un acceso especial (s), permitiendo que un ejecutable se inicie con los permisos del propietario del archivo en lugar de los permisos del usuario que lo ejecuta, permitiendo configuraciones anómalas, la posibilidad de elevar privilegios a root. [↑](#footnote-ref-37)
39. repositorio en línea que recopila binarios de Unix y Linux que pueden ser utilizados de forma maliciosa para escalada de privilegios, evasión de restricciones de seguridad o ejecución de comandos de manera no autorizada. [↑](#footnote-ref-38)
40. Información del archivo crontab, donde se pueden programar tareas, existiendo el archivo “cleanup” que se ejecuta cada minuto con privilegios de usuario “root”, pudiendo ser copiado por el usuario “ww-data” al directorio /tmp para su modificación. [↑](#footnote-ref-39)
41. intérprete de comandos usado en sistemas operativos Unix y Linux. Permite a los usuarios ejecutar comandos, escribir scripts automatizados y gestionar archivos y procesos. Bash es una de las shells más populares y se utiliza tanto para la interacción directa con el sistema como para automatizar tareas repetitivas mediante scripts. [↑](#footnote-ref-40)
42. archivo “cleanup” malicioso con un verificación de ejecución y una reverse shell de python que sustituirá al original en la ruta de ejecución del crontab. [↑](#footnote-ref-41)
43. Proceso de sustitución del archivo malicioso por el original, permitiéndole tener permisos de ejecución. [↑](#footnote-ref-42)
44. Resultado positivo en el proceso de elevación de privilegios, consiguiendo una reverse shell con máximos privilegios en el sistema. [↑](#footnote-ref-43)
45. 5 hashes de tipo SHA-512 perteneciente a los usuarios autorizados en el sistema [↑](#footnote-ref-44)
46. Resultado del descifrado del hash en SHA-512, del usuario “anne”. [↑](#footnote-ref-45)
47. Directorio en sistemas Unix y Linux que contiene scripts de inicio que se ejecutan automáticamente cuando el sistema arranca o se apaga, controlando el inicio, parada y reinicio de servicios y procesos esenciales del sistema. [↑](#footnote-ref-46)
48. resultado del envío del script malicioso de nuestra Kali a la maquina objetivo [↑](#footnote-ref-47)
49. LSB (Linux Standard Base) en un script es una sección que proporciona metadatos sobre el mismo, especialmente en los scripts de inicio en sistemas Unix/Linux, siguiendo un formato estándar, que se usa para definir información como la descripción del servicio, dependencias, niveles de ejecución, y órdenes de inicio y parada, facilitando la gestión de servicios en el sistema. [↑](#footnote-ref-48)
50. Script del creación del servicio utilizado para obtener persistencia, activando solo la fase de inicio, que es la que nos interesa, con niveles de ejecución: 2 (multiusuario sin soporte para redes), 3 (multiusuario con red en modo CLI), 4(multiusuario para casos especiales) y 5(multiusuario con red y entorno gráfico. Ademas es ha establecido en una variable donde se halla el script a ejecutar con su cargador php, condicionando el inicio del script , el cual se iniciara en segundo plano en caso de cumplirse el argumento. [↑](#footnote-ref-49)
51. se observa como el servicio malicioso “reto\_16” se halla en /etc/init.d con permisos de ejecución, ejecutando el comando : update-rc.d reto\_16 defaults, agregándose de manera satisfactoria [↑](#footnote-ref-50)
52. EDR.- solución de seguridad centrada en la detección y respuesta ante amenazas en los dispositivos finales, mientras que el XDR, amplia el concepto anterior, incluyendo áreas como la red, servidores, y aplicaciones, permitiendo una detección, correlación y respuesta a nivel más amplio y coordinado. [↑](#footnote-ref-51)