18 de junio de 2021
Ficha N° 8 UNISCAN
CSIRT DE GOBIERNO

Comando de la semana "UNISCAN"

I. CONTEXTO

Este documento, denominado "comando de la semana", tiene como objetivo ilustrar sobre herramientas que pueden ser de utilidad para el lector, a objeto de ir potenciando las capacidades locales de autochequeo, detección simple de vulnerabilidades que están expuestas a internet en sus activos de información y, a su vez, la obtención de una verificación de la subsanación de aquellas que se les han sido reportadas, facilitando la interacción con el CSIRT de Gobierno. El objetivo no es reemplazar una auditoria de código o evaluación de vulnerabilidades, sino que establecer capacidades básicas de chequeo y obtención de información de manera rápida para temas específicos, como por ejemplo la verificación de la subsanación de alertas o vulnerabilidades reportadas por "CSIRT GOB CL". Todas estas herramientas al contar con la posibilidad de ser usadas desde una línea de comando permiten en algún grado la integración dentro de script o lenguajes de automatización o programación como PERL, AWK, Shell Scripting[1], Expect, Python, C, C++, Golang, JavaScript, PowerShell, Ruby, Java, PHP, Elixir, Elm, Go, Dart, Pony, TypeScript, Kotlin, Nim, OCaml, Reason, Rust, entre otros con miras a automatizar estas actividades y concentrar el tiempo de los especialistas en el análisis de los datos para encontrar los problemas relevantes y descartar los falsos positivos.

II. INTRODUCCIÓN

Una de las tareas regulares que en ciberseguridad se realizan es la verificación de los sitios o sistemas que están expuestos a Internet. En general uno de los problemas de las herramientas de seguridad es la gran cantidad de falsos positivos que muestran, dificultando concentrar los esfuerzos en los hallazgos realmente críticos. En esta tarea ayuda tener herramientas con funcione similares que ayuden a comparar los resultados obtenidos por otros medios.

Para este caso existen comandos o herramientas en sobre el sistema operativo Linux que nos ayuda a recopilar información manera simple y comparar sus resultados con otras a modo de verificación,

[1] https://scis.uohyd.ac.in/~apcs/itw/UNIXProgrammingEnvironment.pdf

con una herramienta de código abierto y, en base a sus resultados tomar decisiones de mitigación[2], monitoreo y vigilancia. Una de esas herramientas es: UNISCAN.

¿Qué es UNISCAN?

Uniscan es un simple escáner de vulnerabilidad de inclusión de archivos remotos, inclusión de archivos locales y ejecución de comandos remotos.

Que aspectos ayuda a evaluar:

- Backup Files: Busqueda de posibles archivos de respaldo expuestos d emanera abierta dentro de la estructura del sitio o sistema web. Es absolutamente desaconsejable exponer archivos con respaldos de información o del sistema mismo de manera abierta y sin controles de seguridad.
- Blind SQL Injection: La inyección ciega (blind) de SQL (lenguaje de consulta estructurado) es un tipo de ataque de inyección SQL que hace preguntas verdaderas o falsas a la base de datos y determina la respuesta en función de la respuesta de la aplicación. Este ataque se usa a menudo cuando la aplicación web está configurada para mostrar mensajes de error genéricos, pero no ha mitigado el código que es vulnerable a la inyección de SQL. Cuando un atacante aprovecha la inyección de SQL, generalmente lo hace en las situaciones que la aplicación web muestra mensajes de error de la base de datos, alertando que la sintaxis de la consulta SQL es incorrecta. La inyección SQL ciega es casi idéntica a la inyección SQL normal, con la sola diferencia en cuanto a la forma en que se recuperan los datos de la base de datos. Cuando la base de datos no envía datos a la página web, un atacante se ve obligado a robar datos haciendo a la base de datos una serie de preguntas verdaderas o falsas. Esto hace que la explotación de la vulnerabilidad de inyección SQL sea más difícil, pero no imposible. Vea algunos ejemplos en la referencia[3].
- Local File Include: LFI ocurre cuando una aplicación usa la ruta a un archivo como entrada. Si la aplicación trata esta entrada como confiable, se puede usar un archivo local en la declaración de inclusión. La inclusión de archivos locales es muy similar a la inclusión de archivos remotos (RFI) . Sin embargo, un atacante que utilice LFI solo puede incluir archivos locales (no archivos remotos como en el caso de RFI). Un atacante puede utilizar la inclusión de archivos locales (LFI) para engañar a la aplicación web para que exponga o ejecute archivos en el servidor web. Un ataque LFI puede provocar la divulgación de información, la ejecución remota de código o incluso Cross-site Scripting (XSS). Vea algunos ejemplos en la referencia[4].

[2] https://blog.shekyan.com/2011/11/how-to-protect-against-slow-http-attacks.html
[3] https://owasp.org/www-community/attacks/Blind_SQL_Injection
[4] https://www.acunetix.com/blog/articles/local-file-inclusion-lfi/

- PHP CGI Argument Injection: Esto indica un intento de ataque contra una vulnerabilidad de inyección de argumentos en PHP CGI. La vulnerabilidad es causada por un error cuando el software vulnerable maneja una solicitud maliciosa. Permite que un atacante remoto ejecute código arbitrario a través de un URI diseñado. Dettales complementarios en la referencia[5].
- Remote Command Execution: Según OWASP, Code Injection es el término general para los tipos de ataque que consisten en inyectar código que luego es interpretado / ejecutado por la aplicación. Este tipo de ataque aprovecha el mal manejo de datos que no son de confianza. Estos tipos de ataques suelen ser posibles debido a la falta de una validación adecuada de los datos de entrada / salida. RCE son las siglas de Remote Code Execution (ejecución remota de código). Eso permite a un atacante inyectar su propio código de forma remota en la aplicación de destino. Depende totalmente de las tecnologías backend. PHP, JAVA, ASP.NET, Python son algunas de las tecnologías de backend más populares. Los ataques de inyección de comandos son posibles cuando una aplicación pasa datos no seguros proporcionados por el usuario (formularios, cookies, encabezados HTTP, etc.) a un shell del sistema. En este ataque, los comandos del sistema operativo proporcionados por el atacante generalmente se ejecutan con los privilegios de la aplicación vulnerable. Los ataques de inyección de comandos son posibles en gran parte debido a una validación de entrada insuficiente. La capacidad de ejecutar comandos del sistema de forma remota en la aplicación vulnerable denominada ejecución remota de comandos. Más detalles en la referencia[6].
- Remote File Include: La inclusión remota de archivos (RFI) es una técnica de ataque utilizada para explotar los mecanismos de "inclusión dinámica de archivos" en aplicaciones web. Cuando las aplicaciones web toman la entrada del usuario (URL, valor de parámetro, etc.) y las pasan a los comandos de inclusión de archivos, la aplicación web puede ser engañada para que incluya archivos remotos con código malicioso. Casi todos los marcos de aplicaciones web admiten la inclusión de archivos. La inclusión de archivos se usa principalmente para empaquetar código común en archivos separados que luego son referenciados por los módulos de la aplicación principal. Cuando una aplicación web hace referencia a un archivo de inclusión, el código de este archivo puede ejecutarse implícita o explícitamente llamando a procedimientos específicos. Si la elección del módulo para cargar se basa en elementos de la solicitud HTTP, la aplicación web puede ser vulnerable a RFI. Un atacante puede usar RFI para: (i) Ejecutar código malicioso en el servidor: el servidor ejecutará cualquier código de los archivos maliciosos incluidos. Si el archivo de inclusión no se ejecuta utilizando algún contenedor, el código de los archivos de inclusión se ejecuta en el contexto del usuario del servidor. Esto podría conducir a un compromiso total del sistema. (ii) Ejecutar código malicioso en los clientes: El código malicioso del atacante puede manipular el contenido de la respuesta enviada al cliente. El atacante puede incrustar código

[5] https://www.rapid7.com/db/modules/exploit/multi/http/php_cgi_arg_injection/
[6] https://owasp.org/www-community/attacks/Command_Injection

malicioso en la respuesta que ejecutará el cliente (por ejemplo, JavaScript para robar las cookies de sesión del cliente). PHP es particularmente vulnerable a los ataques de RFI debido al uso extensivo de "archivos incluidos" en la programación PHP y debido a las configuraciones de servidor predeterminadas que aumentan la susceptibilidad a un ataque de RFI. Más detalles en la referencia[7].

- SQL Injection: Un ataque de inyección SQL consiste en la inserción o "inyección" de una consulta SQL a través de los datos de entrada del cliente a la aplicación. Un exploit de inyección SQL exitoso puede leer datos confidenciales de la base de datos, modificar los datos de la base de datos (Insertar / Actualizar / Eliminar), ejecutar operaciones de administración en la base de datos (como cerrar el DBMS), recuperar el contenido de un archivo dado presente en el archivo DBMS system y, en algunos casos, emitir comandos para el sistema operativo. Los ataques de inyección SQL son un tipo de ataque de inyección, en el que los comandos SQL se inyectan en la entrada del plano de datos para afectar la ejecución de comandos SQL predefinidos.Mas detalles en la referencia[8].
- Cross-Site Scripting (XSS): Los ataques de Cross-Site Scripting (XSS) son un tipo de inyección, en el que se inyectan scripts maliciosos en sitios web que de otro modo serían benignos y confiables. Los ataques XSS ocurren cuando un atacante usa una aplicación web para enviar código malicioso, generalmente en forma de un script del lado del navegador, a un usuario final diferente. Las fallas que permiten que estos ataques tengan éxito están bastante extendidas y ocurren en cualquier lugar donde una aplicación web use la entrada de un usuario dentro de la salida que genera sin validarla o codificarla. Un atacante puede usar XSS para enviar un script malicioso a un usuario desprevenido. El navegador del usuario final no tiene forma de saber que no se debe confiar en el script y lo ejecutará. Debido a que cree que el script proviene de una fuente confiable; el script malicioso puede acceder a las cookies, tokens de sesión u otra información confidencial retenida por el navegador y utilizada con ese sitio. Estos scripts pueden incluso reescribir el contenido de la página HTML. Más detalles en la referencia[9].
- Web Shell Finder: La mayoría de los servidores web ejecutan código del lado del servidor que se procesa (interpreta) en tiempo de ejecución cuando el navegador web de un cliente solicita la URL que apunta a ese código. Esto incluye servidores web basados en PHP, JSP y ASP, por ejemplo. En lugar de haber una aplicación de software monolítica que responda a cada solicitud del cliente, generalmente hay un archivo de código que se 'ejecuta' en respuesta a cada URL diferente solicitada por el cliente. Por ejemplo: cuando el navegador del cliente solicita una URL, como: www.mysite.com/myapp/welcome, el servidor web

[7] https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/11.2-Testing_for_Remote_File_Inclusion
[8] https://owasp.org/www-community/attacks/SQL_Injection
[9] https://owasp.org/www-community/attacks/xss/

interpreta / ejecuta el código en el archivo ' bienvenida' y devuelve la respuesta generada por el código al navegador del cliente. En otras palabras, cada vez que se solicita una URL desde el servidor web, puede considerar la solicitud de los navegadores como simplemente una instrucción para "ejecutar este archivo en el servidor". Un ataque de shell web ocurre cuando un usuario malintencionado puede inyectar su propio archivo en el directorio del servidor web para que luego pueda indicarle al servidor web que ejecute ese archivo simplemente solicitándolo desde su navegador web. Los Web Shells se instalan en los servidores aprovechando las vulnerabilidades del software del servidor web o los complementos vulnerables que se agregan al servidor web (por ejemplo plugins de un wordpress). Más detalles en la referencia[10].

NOTA: Dado que es importante un buen manejo de los comandos básicos de Linux, tanto para posteriores manipulaciones como para usos de la información resultante de la ejecución de los comandos, es que el comité editorial decidió que se incluya en esta edición y en las subsiguientes un anexo de comandos Linux que son de utilidad para moverse en este sistema operativo. Se sugiere dominarlos todos para facilitar el acceso y manipulación de la información. En futuras ediciones se irán incorporando nociones más avanzadas sobre el uso de estos comandos para procesamiento de archivos, procesos, y de sus usos en scripting.

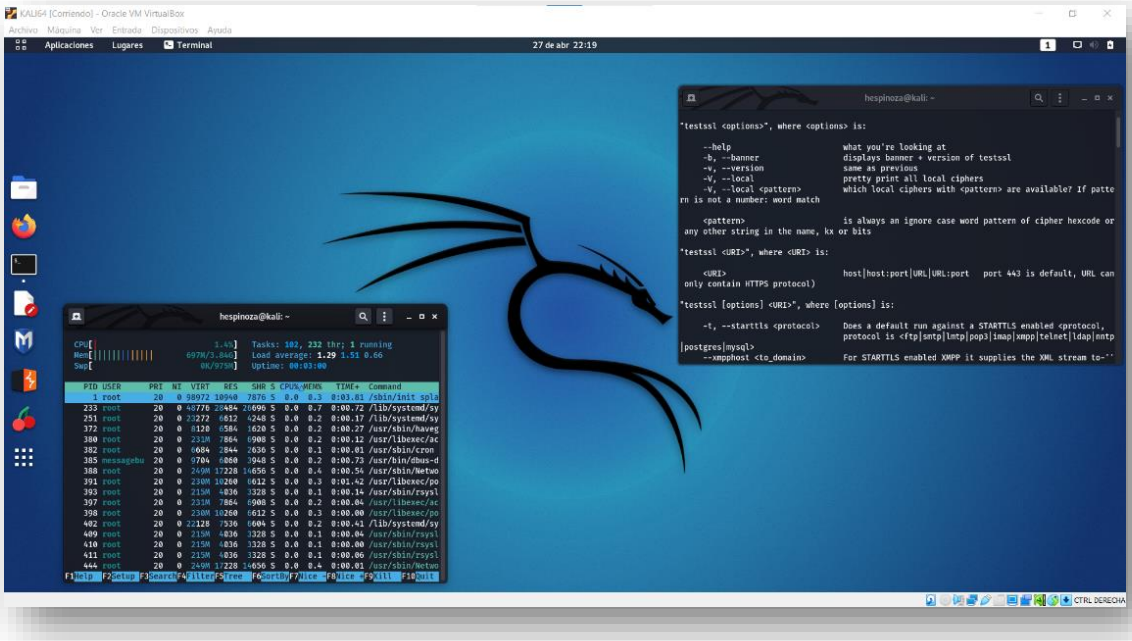
Vea anexo I: Comandos básicos de Linux

[10] https://secureteam.co.uk/articles/web-application-security-articles/what-are-web-shell-attacks/

III. PASO A PASO

PASO 1: Un entorno adecuado para trabajar.

Primero debe contar con una distribución de Kali[11] Linux funcionando ya sea en una máquina física o en una máquina virtual[12][13].



[11] https://www.kali.org/downloads/

[12] https://my.vmware.com/en/web/vmware/downloads/info/slug/desktop_end_user_computing/vmware_workstation_player/16_0

[13] https://www.virtualbox.org/wiki/Downloads

PASO 2: Instalar el comando.

Una vez que se cuenta con este sistema operativo de manera funcional podemos instalar el comando "UNISCAN"; en general este ya viene preinstalado en la distribución KALI[14], pero si no fuere así puede instalarlo con los siguientes comandos, **previamente tomando privilegios de usuario "root"**:

```
#apt-get install uniscan
```

Un avez que esté instalado debiera ver algo como lo siguiente al buscar el paquete dentro de su equipo:

```
#apt search ^uniscan
Ordenando... Hecho
Buscar en todo el texto... Hecho
uniscan/kali-rolling,now 6.3-0kali2 all [instalado, automático]
  LFI, RFI, and RCE vulnerability scanner
```

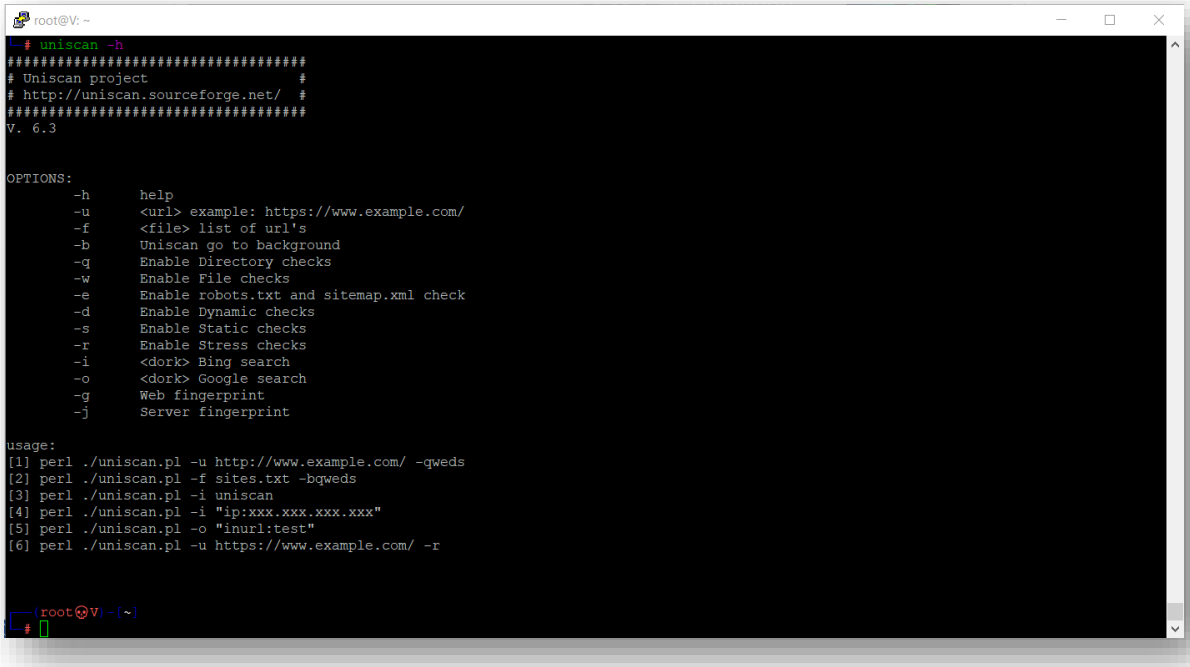
Nota: El símbolo "^" le instruye a la orden de búsqueda que encuentre solo los paquetes que comiencen por la palabra "uniscan" en este ejemplo.

[14] https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-image-download/

PASO3: Verificar su instalación.

Una vez que se instalado podemos verificar y explorar las múltiples opciones que ofrece para su ejecución:

En una consola de su KALI ejecute el comando para que muestre la ayuda: “uniscan -h”.



Debiéramos lograr desplegar todas las opciones y parámetros de ejecución, junto a su explicación en la consola.

```
# uniscan -h
###################################
# Uniscan project                 #
# http://uniscan.sourceforge.net/  #
###################################
V. 6.3


OPTIONS:
        -h      help
        -u      <url> example: https://www.example.com/
        -f      <file> list of url's
        -b      Uniscan go to background
        -q      Enable Directory checks
```

```
        -w      Enable File checks
        -e      Enable robots.txt and sitemap.xml check
        -d      Enable Dynamic checks
        -s      Enable Static checks
        -r      Enable Stress checks
        -i      <dork> Bing search
        -o      <dork> Google search
        -g      Web fingerprint
        -j      Server fingerprint

usage:
[1] uniscan -u http://www.example.com/ -qweds
[2] uniscan -f sites.txt -bqweds
[3] uniscan -i uniscan
[4] uniscan -i "ip:xxx.xxx.xxx.xxx"
[5] uniscan -o "inurl:test"
[6] uniscan -u https://www.example.com/ -r
```

Paso 4: Ponerlo en marcha para verificar nuestra infraestructura.

Un ejemplo de ejecución básica para nuestros primeros pasos:

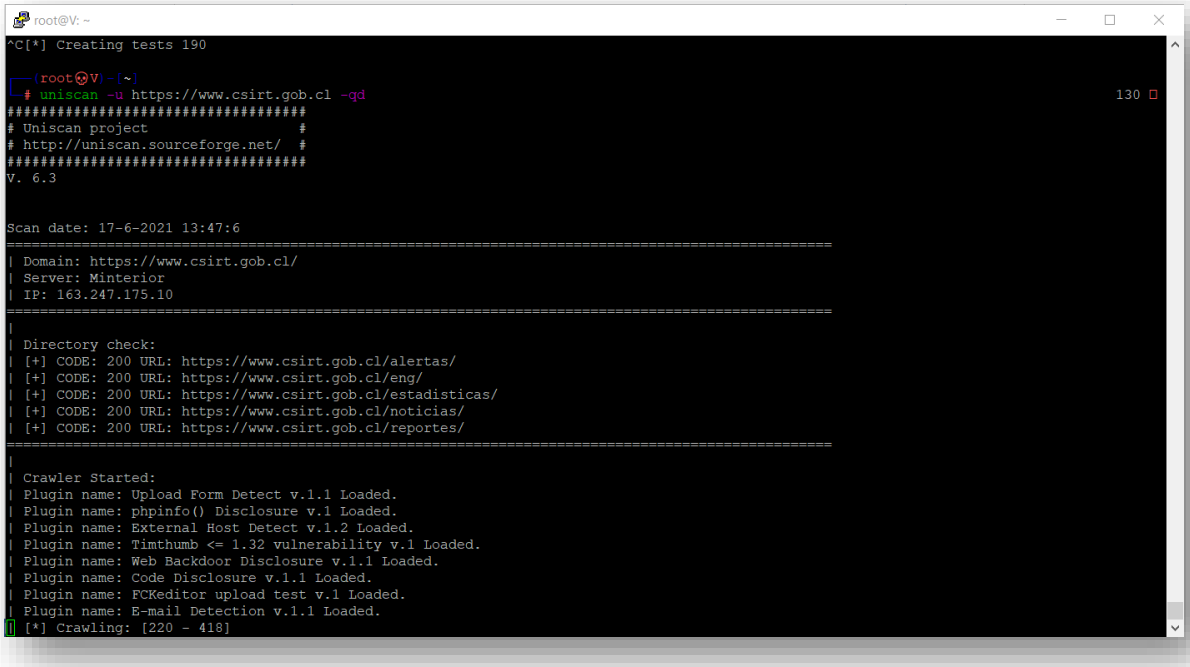
Probaremos el sitio web https://www.csirt.gob.cl.

```
                                    EJEMPLO


root@kali:~# uniscan -u https://www.csirt.gob.cl -qd

Nota:
-u      <url> example: https://www.example.com/
-q      Enable Directory checks
-d      Enable Dynamic checks
```

Al ejecutar el comando se puede observar en la consola lo siguiente:



Una vez que ha finalizado el test, despúes d evarios minutos de procesamiento y trabajo, indica lo siguiente:

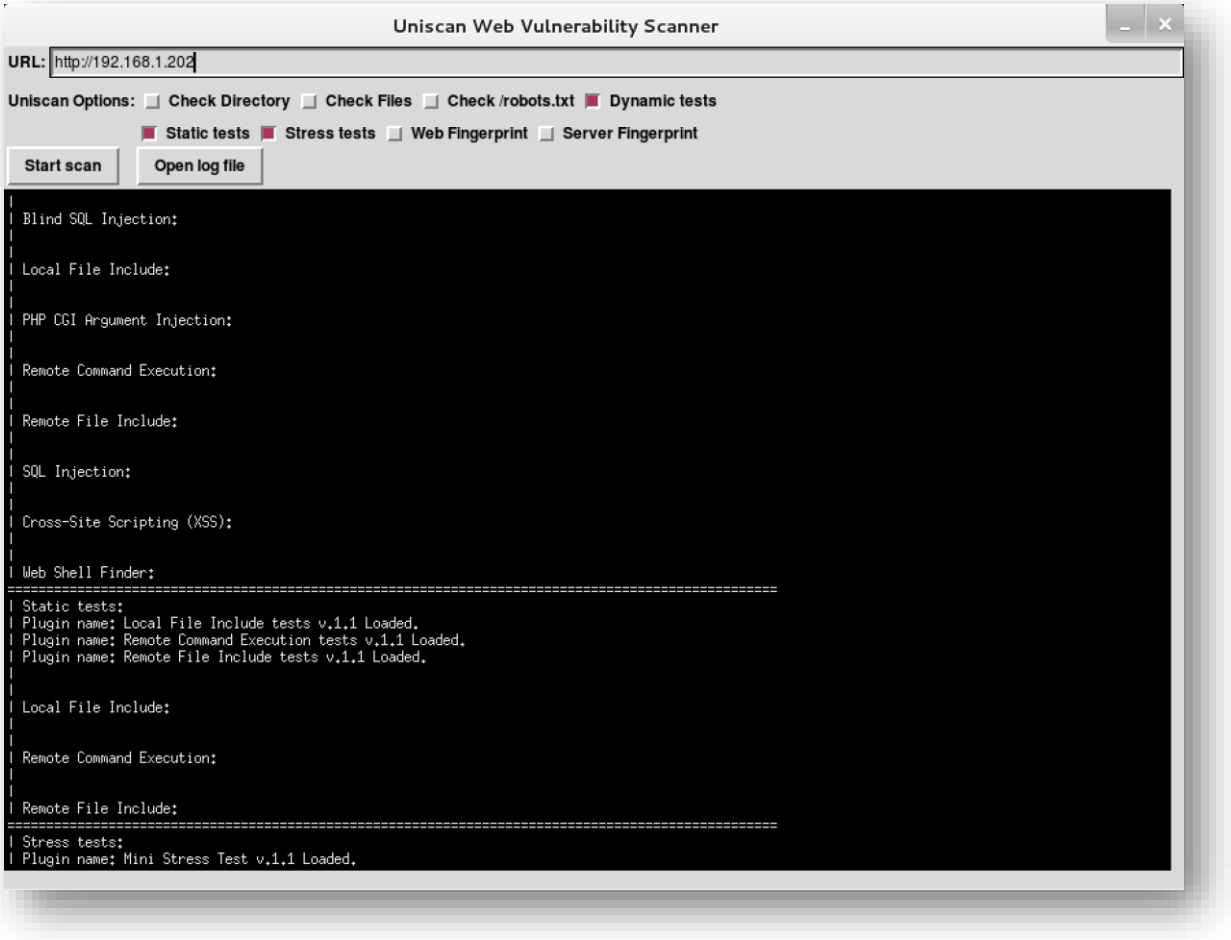
Como producto de la ejecución se generan dos archivos:

- Deja un archivo en formato “HTML” el cual puede ser visualizado con un navegador web.

Eventualmente para efectos de facilidad de uso inicial, uniscan ofrece una interfaz gráfica, pero se aconseja preferir la opción de línea de comando para manejar con precisión las opciones y poder así integrarlo en futuras implementaciones automatizadas.

EJEMPLO USO INTERFAZ GRÁFICA

```
root@kali:~# uniscan-gui
```



Tenga presente que es importante que estas pruebas sean coordinadas con el equipo de operaciones y en ambientes que estén bajo supervisión, pues eventualmente si el sitio o sistema web está mal construido o implementado, puede producirse algún tipo de afectación del servicio e impactar a sus usuarios.

Defina horarios especiales o ambientes de "test o QA" equivalentes a los de "producción", para mitigar los posibles efectos perjudiciales en el sitio o sistema web.

Estudie las múltiples opciones que tiene el comando para obtener resultados específicos o redirigir la salida a un archivo, para su inclusión en informes posteriores.

En caso de cualquier inquietud no dude en consultarnos a soc-csirt@interior.gob.cl.

Si encuentra algún error en el documento también es importante que nos lo comunique para introducir las correcciones pertinentes en las versiones futuras de esta ficha.

Anexo I: Comandos Básicos de Linux

Comandos básicos

Los comandos son esencialmente los mismos que cualquier sistema UNIX. En las tablas que se presentan a continuación se tiene la lista de comandos más frecuentes.

Comando/Sintaxis	**Descripción**	**Ejemplos**
cat *fich1* [*...fichN*]	Concatena y muestra un archivos	cat /etc/passwd
	archivos	cat dict1 dict2 dict
cd [*dir*]	Cambia de directorio	cd /tmp
chmod *permisos fich*	Cambia los permisos de un archivo	chmod +x miscript
chown *usuario:grupo fich*	Cambia el dueño un archivo	chown nobody miscript
cp *fich1...fichN dir*	Copia archivos	cp foo foo.backup
diff [*-e*]*arch1 arch2*	Encuentra diferencia entre archivos	diff foo.c newfoo.c
du [*-sabr*] *fich*	Reporta el tamaño del directorio	du -s /home/
file *arch*	Muestra el tipo de un archivo	file arc_desconocido
find *dir test acción*	Encuentra archivos.	find . -name ``.bak" –print
grep [*-cilnv*] *expr archivos*	Busca patrones en archivos	grep mike /etc/passwd
head *-count fich*	Muestra el inicio de un archivo	head prog1.c
mkdir *dir*	Crea un directorio.	mkdir temp
mv *fich1 ...fichN dir*	Mueve un archivo(s) a un directorio	mv a.out prog1
mv *fich1 fich2*	Renombra un archivo.	mv .c prog_dir
less / more *fich(s)*	Visualiza página a página un archivo.	more muy_largo.c
	less acepta comandos vi.	less muy_largo.c
ln [*-s*] *fich acceso*	Crea un acceso directo a un archivo	ln -s /users/mike/.profile .

ls	Lista el contenido del directorio	ls -l /usr/bin
pwd	Muestra la ruta del directorio actual	Pwd
rm *fich*	Borra un fichero.	rm foo.c
rm -r *dir*	Borra un todo un directorio	rm -rf prog_dir
rmdir *dir*	Borra un directorio vacío	rmdir prog_dir
tail -*count fich*	Muestra el final de un archivo	tail prog1.c
vi *fich*	Edita un archivo.	vi .profile

Comandos Linux/Unix de manipulación de archivos y directorios:

Comando/Sintaxis	**Descripción**	**Ejemplos**
at [-*lr*] *hora* [*fecha*]	Ejecuta un comando mas tarde	at 6pm Friday miscript
cal [[*mes*] *año*]	Muestra un calendario del mes/año	cal 1 2025
date [*mmddhhmm*] [+*form*]	Muestra la hora y la fecha	Date
echo *string*	Escribe mensaje en la salida estándar	echo ``Hola mundo''
finger *usuario*	Muestra información general sobre un usuario en la red	finger nn@maquina.aca.com.co
id	Número id de un usuario	id usuario
kill [-*señal*] *PID*	Matar un proceso	kill 1234
man *comando*	Ayuda del comando especificado	man gcc man -k printer
passwd	Cambia la contraseña.	passwd
ps [*axiu*]	Muestra información sobre los procesos que se están ejecutando en el sistema	ps -ux
who / rwho	Muestra información de los usuarios conectados al sistema.	who

Comandos Linux/Unix más frecuentes:

Linux	DOS	Significado
cat	type	Ver contenido de un archivo.
cd, chdir	cd, chdir	Cambio el directorio en curso.
chmod	attrib	Cambia los atributos.
clear	cls	Borra la pantalla.
ls	dir	Ver contenido de directorio.
mkdir	md, mkdir	Creación de subdirectorio.
more	more	Muestra un archivo pantalla por pantalla.
mv	move	Mover un archivo o directorio.
rmdir	rd, rmdir	Eliminación de subdirectorio.
rm -r	deltree	Eliminación de subdirectorio y todo su contenido.