20 de mayo de 2021
Ficha N° 4 GOLISMERO
CSIRT DE GOBIERNO

Comando de la semana “GOLISMERO”

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.

II. Introducción

¿Qué hacer si desde el CSIRT nos llega un ticket señalando que hay problemas con la seguridad de algún sitio o sistema web? ¿Cómo verificamos, una vez que hemos aplicado alguna mitigación y queremos probar si ha tenido efecto, antes de reportarla como “problema solucionado” al CSIRT o a nuestros auditores internos?

Para este caso existe un comando Linux que nos ayuda a detectar algunas vulnerabilidades de una manera simple, con una herramienta de código abierto y, en base a sus resultados tomar decisiones de control de acceso, verificación de mitigación u otras estrategias de resolución de problemas: GOLISMERO.

¿Qué es GOLISMERO?

GoLismero es un framework de código abierto para pruebas de seguridad. Actualmente está orientado a la seguridad web, pero puede ampliarse fácilmente a otros tipos de análisis.

Las características más interesantes del framework son:

- Independencia real de la plataforma. Probado en Windows, Linux, *BSD y OS X.
- Sin dependencias de librerías nativas. Todo el framework ha sido escrito en Python puro.

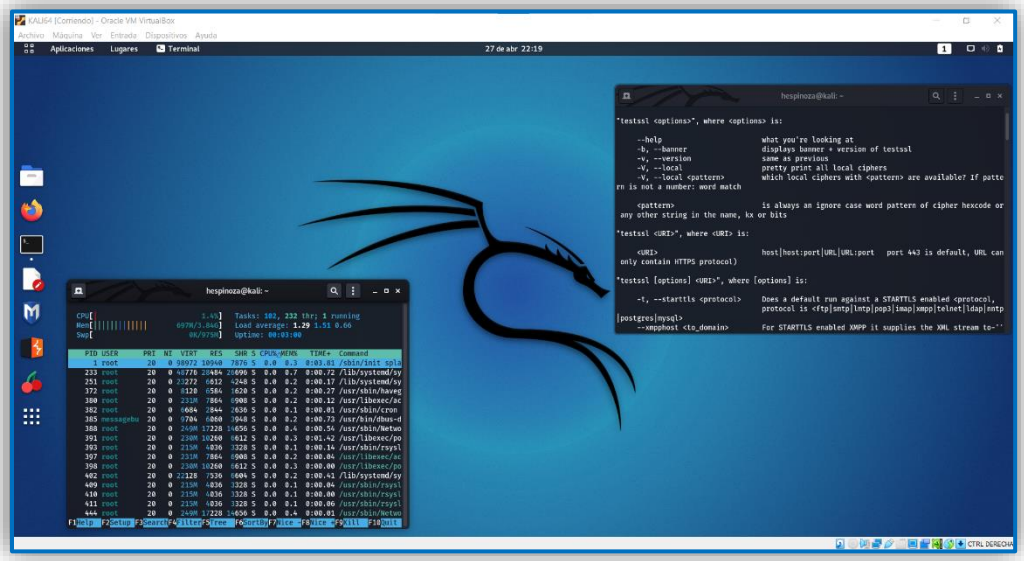
- Buen rendimiento en comparación con otros frameworks escritos en Python y otros lenguajes de scripting.
- Muy fácil de usar.
- El desarrollo de plugins es extremadamente sencillo.
- El framework también recoge y unifica los resultados de herramientas muy conocidas: sqlmap, xsser, openvas, dnsrecon, theharvester...
- Integración con los estándares: CWE, CVE y OWASP.

Fuente: https://github.com/golismero/golismero

III. Paso a Paso

PASO 1: Un entorno adecuado para trabajar.

Primero debe contar con una distribución de Kali[1] Linux funcionando ya sea en una máquina física o en una máquina virtual[2][3].



PASO 2: Instalar el comando.

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

```
apt-get install python2.7 python2.7-dev python-pip python-docutils git perl nmap sslscan
cd /opt
git clone https://github.com/golismero/golismero.git
cd golismero
pip install -r requirements.txt
pip install -r requirements_unix.txt
ln -s ${PWD}/golismero.py /usr/bin/golismero
```

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

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

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

Si tiene una clave API para Shodan, o un servidor OpenVAS o SpiderFoot que quiera integrar con GoLismero, ejecute los siguientes comandos:

```
mkdir ~/.golismero
touch ~/.golismero/user.conf
chmod 600 ~/.golismero/user.conf
nano ~/.golismero/user.conf
```

El último comando es un editor (nano) que despliega el contenido del archivo creado “user.conf” y permite editarlo para incorporar las llaves (API key) de servicios externos para integrarlos al análisis:

```
[shodan:Configuration]
apikey = <INSERT YOUR SHODAN API KEY HERE>

[openvas]
host = <INSERT THE OPENVAS HOST HERE>
user = <INSERT THE OPENVAS USERNAME HERE>
*password = <INSERT THE OPENVAS PASSWORD HERE>

[spiderfoot]
url = <INSERT THE SPIDERFOOT URL HERE>
```

PASO3: Verificar su instalación.

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

```
golismero -h

/-----------------------------------------\
| GoLismero 2.0.0b6, The Web Knife        |
| Copyright (C) 2011-2014 GoLismero Project |
|                                         |
| Contact: contact@golismero-project.com  |
\-----------------------------------------/

usage: golismero COMMAND [TARGETS...] [--options]

 SCAN:
  Perform a vulnerability scan on the given targets. Optionally import
  results from other tools and write a report. The arguments that follow may
  be domain names, IP addresses or web pages.

 RESCAN:
  Same as SCAN, but previously run tests are repeated. If the database is
  new, this command is identical to SCAN.

 PROFILES:
  Show a list of available config profiles. This command takes no arguments.

 PLUGINS:
  Show a list of available plugins. This command takes no arguments.

 INFO:
  Show detailed information on a given plugin. The arguments that follow are
  the plugin IDs. You can use glob-style wildcards.

 REPORT:
  Write a report from an earlier scan. This command takes no arguments.
  To specify output files use the -o switch.

 IMPORT:
  Import results from other tools and optionally write a report, but don't
  scan the targets. This command takes no arguments. To specify input files
  use the -i switch.

 DUMP:
  Dump the database from an earlier scan in SQL format. This command takes no
```

```
  arguments. To specify output files use the -o switch.

 LOAD:
  Load a database dump from an earlier scan in SQL format. This command takes
  no arguments. To specify input files use the -i switch.

 UPDATE:
  Update GoLismero to the latest version. Requires Git to be installed and
  available in the PATH. This command takes no arguments.
```

Listado de PLUGINS disponibles para nuestros análisis:

```
┌──(root💀kali)-[~]
└─# golismero plugins

/-------------------------------------------\
| GoLismero 2.0.0b6, The Web Knife          |
| Copyright (C) 2011-2014 GoLismero Project |
|                                           |
| Contact: contact@golismero-project.com    |
\-------------------------------------------/

-------------
 Plugin list
-------------

-= Import plugins =-

csv_nikto:
   Import the results of a Nikto scan in CSV format.

csv_spiderfoot:
   Import the results of a SpiderFoot scan in CSV format.

xml_nmap:
   Import the results of an Nmap scan in XML format.

xml_openvas:
   Import the results of an OpenVAS scan in XML format.

xml_sslscan:
   Import the results of an SSLScan run in XML format.
```

-= Recon plugins =-

dns:
DNS resolver plugin.
Without it, GoLismero can't resolve domain names to IP addresses.

dns_malware:
Detect if a domain has been potentially spoofed, hijacked.

exploitdb:
Integration with Exploit-DB (http://www.exploit-db.com/)
This plugin requires a working Internet connection to run.

fingerprint_web:
Fingerprinter of web servers.

punkspider:
Integration with PunkSPIDER (http://punkspider.hyperiongray.com/)
This plugin requires a working Internet connection to run.

robots:
Analyzes robots.txt files and extracts their links.

shodan:
Integration with Shodan: http://www.shodanhq.com/
This plugin requires a working Internet connection to run.

spider:
Web spider plugin.
Without it, GoLismero can't crawl web sites.

spiderfoot:
Integration with SpiderFoot: http://www.spiderfoot.net/

theharvester:
Integration with theHarvester: https://github.com/MarioVilas/theHarvester/

-= Scan plugins =-

brute_directories:
Tries to discover hidden folders by brute force:
www.site.com/folder/ -> www.site.com/folder2 www.site.com/folder3 ...

brute_dns:

Tries to find hidden subdomains by brute force.

brute_url_extensions:
Tries to discover hidden files by brute force:
www.site.com/index.php -> www.site.com/index.php.old

brute_url_permutations:
Tries to discover hidden files by bruteforcing the extension:
www.site.com/index.php -> www.site.com/index.php2

brute_url_predictables:
Tries to discover hidden files at predictable locations.
For example: (Apache) www.site.com/error_log

brute_url_prefixes:
Tries to discover hidden files by bruteforcing prefixes:
www.site.com/index.php -> www.site.com/~index.php

brute_url_suffixes:
Tries to discover hidden files by bruteforcing suffixes:
www.site.com/index.php -> www.site.com/index2.php

nikto:
Integration with Nikto: https://www.cirt.net/nikto2

nmap:
Integration with Nmap: http://nmap.org/

openvas:
Integration with OpenVAS: http://www.openvas.org/

plecost:
WordPress vulnerabilities analyzer, completely rewritten for GoLismero,
based on the original idea of Plecost (https://code.google.com/p/plecost/)
and their team: @ffranz and @ggdaniel

sslscan:
Integration with SSLScan: http://sourceforge.net/projects/sslscan/

zone_transfer:
Detects and exploits DNS zone transfer vulnerabilities.

-= Attack plugins =-

heartbleed:
Test for the CVE-2014-0160 vulnerability (aka "heartbleed attack").

sqlmap:
SQL Injection plugin, using SQLMap.
Only retrieves the DB banner, does not exploit any vulnerabilities.

xsser:
Integration with XSSer: http://xsser.sourceforge.net/

-= Report plugins =-

bson:
BSON (Binary JSON) output for programmatic access.

csv:
Writes reports in Comma Separated Values format.

html:
Writes reports as offline web pages.

json:
JSON output for programmatic access.

latex:
Writes reports in LaTeX document format (.tex).

log:
Extracts only the logs.

ltsv:
Extracts only the logs, in labeled tab-separated values format.

msgpack:
MessagePack output for programmatic access.
See: http://msgpack.org/

odt:
Writes reports in OpenOffice document format (.odt).

rst:
Writes reports in reStructured Text format.

text:

Writes plain text reports to a file or on screen.

xml:
XML output for programmatic access.

yaml:
YAML output for programmatic access.

-= UI plugins =-

console:
Console user interface. This is the default.

disabled:
Empty user interface. Used by some unit tests.

Paso 4: Ponerlo en marcha para verificar nuestra infraestructura.

Algunos ejemplos de ejecución básica para nuestros primeros pasos:

EJEMPLOS

Escanear un sitio web y mostrar los resultados en pantalla:
golismero scan http://www.example.com

En base al resultado del commando NMAP (visto en el episodio anterior de esta serie), escanea todos los host encontrados y escribe el resultado en el archive report.html:
golismero scan -i nmap_output.xml -o report.html

Toma el resultado de un análisis OpenVAS y lo muestra en pantalla, pero sin escanear nada:
golismero import -i openvas_output.xml

Muestra una lista de todos los perfiles de configuración disponibles:
golismero profiles

Muestra una lista de todos los plugins disponibles:
golismero plugins

Muestra información de todos los plugins de fuerza bruta:
golismero info brute_*

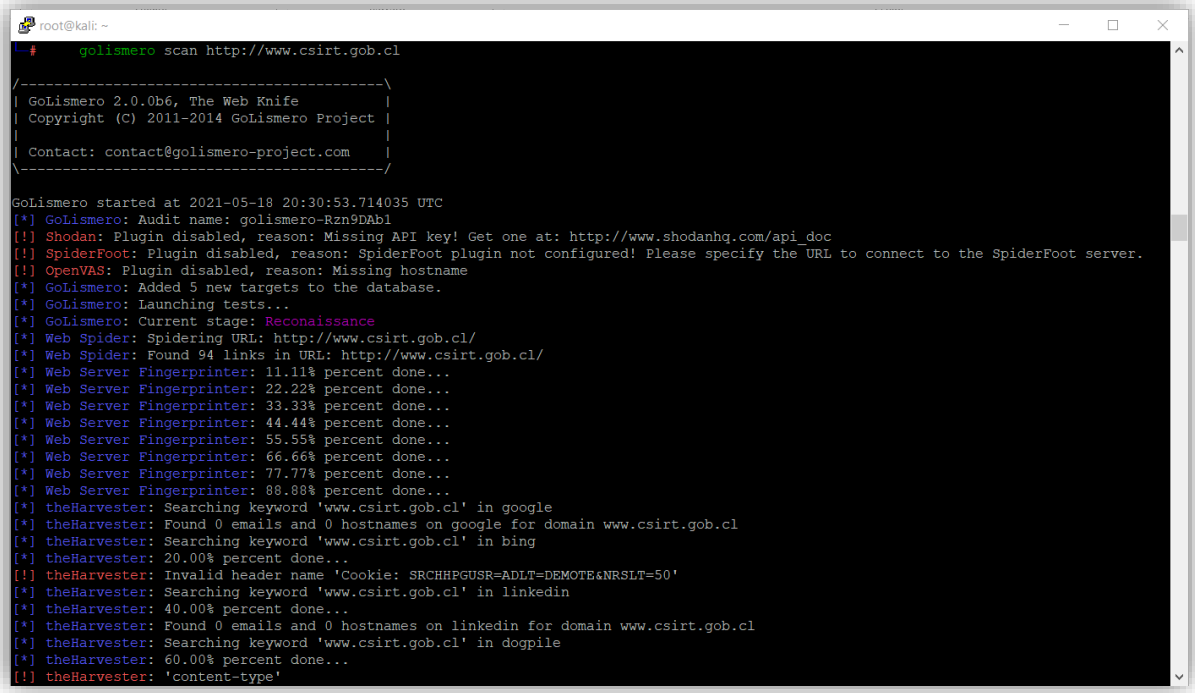
Realiza un dump de base de datos desde un escaneo realizado previamente:
golismero dump -db example.db -o dump.sql

Como se ve un fragmento de reporte en una consola KALI después de la ejecución más simple:

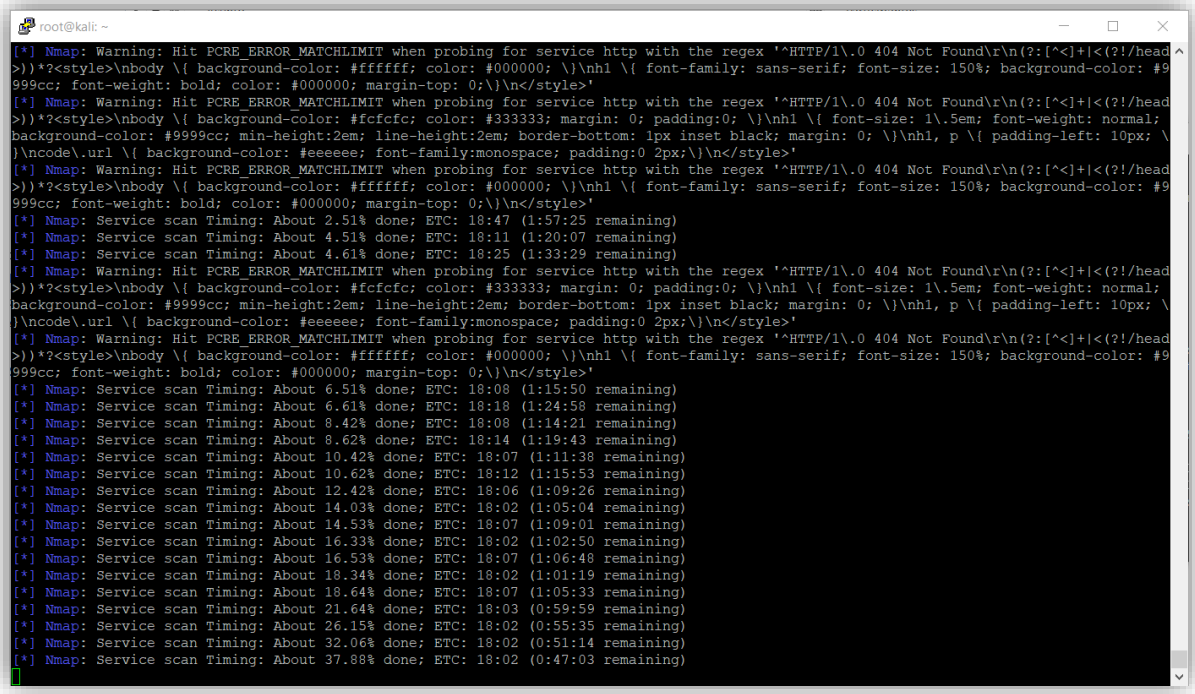
Vista Parcia de un ejemplo:

Ejecución del comando:

#golismero scan http://www.csirt.gob.cl



La ejecución de este comando toma su tiempo, pues se apoya en otros comandos complementarios

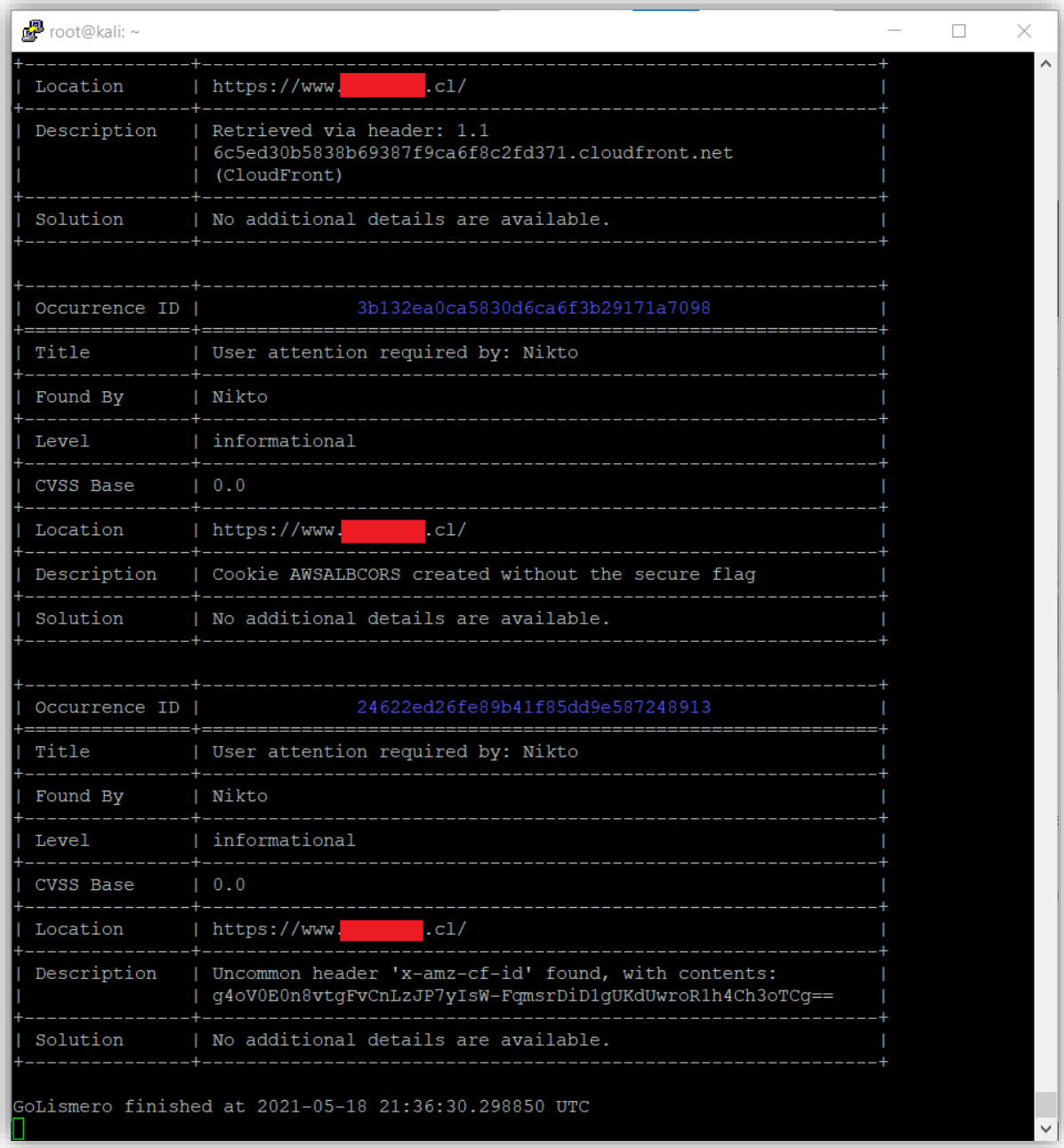


de apoyo al escaneo principal como NMAP, SSLSCAN, entre otros plugins, y va buscando información en el entorno del sitio o sistema web principal.

Una vez que ha finalizado (pueden ser horas de procesamiento, así que paciencia), entrega un reporte en pantalla con los hallazgos encontrados:

Acá ilustramos un par de fragmentos del reporte que a veces es extenso:

```
root@kali: ~
--= Report =--

-# Summary #-

Audit started:   2021-05-18 15:31:52.232874 UTC
Audit ended:     2021-05-18 17:36:30.041814 UTC
Execution time:  0 days, 2 hours, 4 minutes and 37 seconds

Scanned hosts:   14
Vulnerabilities: 22

-# Vulnerabilities #-

-- Domain Disclosure (vulnerability/information_disclosure/domain_disclosure) --


+---------------+-----------------------------------------------------------------+
| Occurrence ID |             b26cec70a9f49b4727c92dd5fdf92fcd                    |
+===============+=================================================================+
| Title         | Possible subdomain leak                                         |
+---------------+-----------------------------------------------------------------+
| Found By      | DNS Bruteforcer                                                 |
+---------------+-----------------------------------------------------------------+
| Level         | low                                                             |
+---------------+-----------------------------------------------------------------+
| CVSS Base     | 2.2                                                             |
+---------------+-----------------------------------------------------------------+
| Location      | autodiscover.[redacted].cl                                      |
+---------------+-----------------------------------------------------------------+
| Description   | A subdomain was discovered which may be an unwanted             |
|               | information disclosure.                                         |
+---------------+-----------------------------------------------------------------+
| Solution      | Please visit the reference website for more information on      |
|               | how to patch this vulnerability.                                |
+---------------+-----------------------------------------------------------------+
| Taxonomy      | CWE-200                                                         |
+---------------+-----------------------------------------------------------------+
| References    | https://cwe.mitre.org/data/definitions/200.html                 |
|               | https://www.owasp.org/index.php/Information_Leakage             |
+---------------+-----------------------------------------------------------------+


+---------------+-----------------------------------------------------------------+
| Occurrence ID |             b012e1c54e0723649cf89e237bb3e226                    |
+===============+=================================================================+
| Title         | Possible subdomain leak                                         |
+---------------+-----------------------------------------------------------------+
| Found By      | DNS Bruteforcer                                                 |
+---------------+-----------------------------------------------------------------+
| Level         | low                                                             |
```



El resultado de este comando puede ser usado como evidencia de verificación para indicar que se han subsanado los problemas reportados por CSIRT.

Estudie las múltiples opciones que tiene el comando para obtener resultados específicos o redirigir la salida de este hacia otros formatos de archivo, para su inclusión en informes posteriores.

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