PRACTICA 2 – TEMA 3

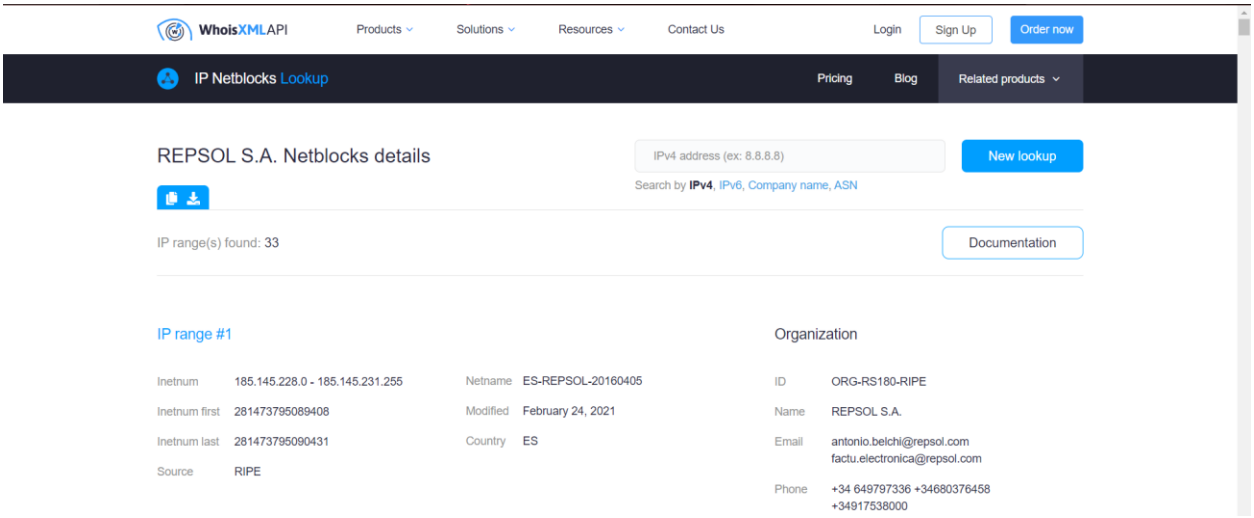
DANIEL KHOMYAKOV TRUBNIKOV

- Desarrollar el proceso completo de identificación de activos para una empresa elegida por el alumno (de aquellas que no hayan sido realizadas en clase).

Para ese apartado hace falta tener una empresa en mente y en mi caso pues lo intentaré con Repsol por lo que lo primero es establecer el método del análisis donde lo que haremos será analizar los sistemas autónomos, rangos de red, dominios y por ultimo los subdominios y se hará en ese orden como se ha especificado. Por ende, lo primero serán los sistemas autónomos donde lo que usaremos será BGP Hurricane

---

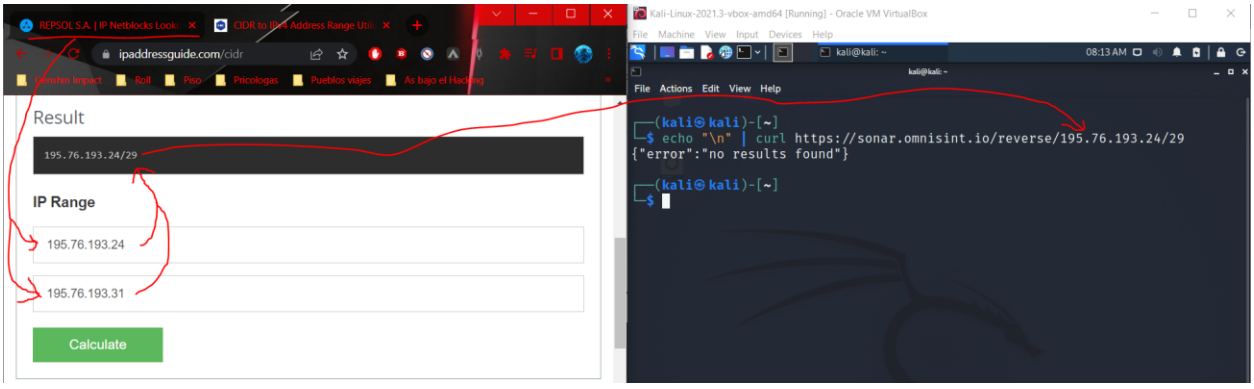
Con esto podemos ver que nos encontramos con los sistemas autónomos de los cuales la primera es la que nos interesa ya que es por la cual empezaré el proceso, aunque luego haré lo mismo con las otras dos. Ahora una vez encuentra estos mismos, ahora miraré el rango de red de REPSOL S.A. y para eso cómo es posible que esta página no pueda darnos todos los rangos de red, para ello miraré mejor en otra como es WhoisXMLapila cual tiene muchas APIs que se pueda usar, pero en mi caso usaré la e IP Netblocks API. Con esta herramienta, con poner le nombre del sistema autónomo pues me dará todos los rangos de red que pueda encontrar esta herramienta, ya que, pues es posible que otra encuentre más o menos, pero esta es bastante viable.



En este caso, nos encuentra hasta 33 rangos de IP de los cuales nos interesa analizarlos todos para encontrar los dominios que se encuentran dentro de estos mismos y en este caso primero sacaré el CIDR para luego usar el SONAR en Kali para sacar los dominios.

```
373  echo "\n"  curl https://sonar.omnisint.io/reverse/212.170.222.56/29
374  echo "\n"  curl https://sonar.omnisint.io/reverse/195.55.121.136/29
375  echo "\n"  curl https://sonar.omnisint.io/reverse/195.55.236.112/29
376  echo "\n"  curl https://sonar.omnisint.io/reverse/195.57.19.72/29
377  echo "\n"  curl https://sonar.omnisint.io/reverse/195.57.90.128/28
378  echo "\n"  curl https://sonar.omnisint.io/reverse/195.77.159.200/29
379  echo "\n"  curl https://sonar.omnisint.io/reverse/195.235.119.112/29
380  echo "\n"  curl https://sonar.omnisint.io/reverse/194.224.85.200/29\n
381  echo "\n"  curl https://sonar.omnisint.io/reverse/194.224.85.200/29
382  echo "\n"  curl https://sonar.omnisint.io/reverse/194.224.109.48/29
383  echo "\n"  curl https://sonar.omnisint.io/reverse/194.224.160.64/27
384  echo "\n"  curl https://sonar.omnisint.io/reverse/195.53.119.112
385  echo "\n"  curl https://sonar.omnisint.io/reverse/195.53.119.112/29
386  echo "\n"  curl https://sonar.omnisint.io/reverse/195.57.157.160/29
387  echo "\n"  curl https://sonar.omnisint.io/reverse/195.76.253.56/29
388  echo "\n"  curl https://sonar.omnisint.io/reverse/213.4.197.224/29
389  echo "\n"  curl https://sonar.omnisint.io/reverse/195.57.19.16/29
390  echo "\n"  curl https://sonar.omnisint.io/reverse/195.57.19.24/29
391  echo "\n"  curl https://sonar.omnisint.io/reverse/195.76.193.24/29
```

Como se puede ver, estos son algunas de las pruebas que hice donde pues todos estos casi siempre me daban error, salvo 4 de estos y para ello hice lo siguiente:



Básicamente lo que hacía era sacar las IPs del WhoisXMLAPI donde luego las metía en la página que se puede ver a la izquierda ya que me sirve para ver el CIDR de estas dos para luego este ser usado en la URL del Sonar, pero por desgracia en este caso no me ha salido casi ningún resultado salvo 4.

```
┌──(kali㉿kali)-[~]
└─$ echo "\n" | curl https://sonar.omnisint.io/reverse/46.25.25.80/29
{"46.25.25.80":["static-80-25-25-46.ipcom.comunitel.net"],"46.25.25.81":["static-81-25-25-46.ipcom.comunitel.net"],"46.25.25.82":["static-82-25-25-46.ipcom.comunitel.net"],"46.25.25.83":["static-83-25-25-46.ipcom.comunitel.net"],"46.25.25.85":["static-85-25-25-46.ipcom.comunitel.net"],"46.25.25.86":["static-86-25-25-46.ipcom.comunitel.net"],"46.25.25.87":["static-87-25-25-46.ipcom.comunitel.net"]}

┌──(kali㉿kali)-[~]
└─$ echo "\n" | curl https://sonar.omnisint.io/reverse/46.25.25.64/28
{"46.25.25.64":["static-64-25-25-46.ipcom.comunitel.net"],"46.25.25.65":["static-65-25-25-46.ipcom.comunitel.net"],"46.25.25.66":["static-66-25-25-46.ipcom.comunitel.net"],"46.25.25.67":["static-67-25-25-46.ipcom.comunitel.net"],"46.25.25.68":["static-68-25-25-46.ipcom.comunitel.net"],"46.25.25.69":["static-69-25-25-46.ipcom.comunitel.net"],"46.25.25.70":["static-70-25-25-46.ipcom.comunitel.net"],"46.25.25.71":["static-71-25-25-46.ipcom.comunitel.net"],"46.25.25.72":["static-72-25-25-46.ipcom.comunitel.net"],"46.25.25.73":["static-73-25-25-46.ipcom.comunitel.net"],"46.25.25.74":["static-74-25-25-46.ipcom.comunitel.net"],"46.25.25.75":["static-75-25-25-46.ipcom.comunitel.net"],"46.25.25.76":["static-76-25-25-46.ipcom.comunitel.net"],"46.25.25.77":["static-77-25-25-46.ipcom.comunitel.net"],"46.25.25.78":["static-78-25-25-46.ipcom.comunitel.net"],"46.25.25.79":["static-79-25-25-46.ipcom.comunitel.net"]}

┌──(kali㉿kali)-[~]
└─$ echo "\n" | curl https://sonar.omnisint.io/reverse/188.85.158.240/29
{"188.85.158.240":["static-240-158-85-188.ipcom.comunitel.net"],"188.85.158.241":["static-241-158-85-188.ipcom.comunitel.net"],"188.85.158.242":["static-242-158-85-188.ipcom.comunitel.net"],"188.85.158.243":["static-243-158-85-188.ipcom.comunitel.net"],"188.85.158.244":["static-244-158-85-188.ipcom.comunitel.net"],"188.85.158.247":["static-247-158-85-188.ipcom.comunitel.net"]}

┌──(kali㉿kali)-[~]
└─$ echo "\n" | curl https://sonar.omnisint.io/reverse/195.53.85.248/29
{"195.53.85.250":["vpnf5.repsol.com"],"195.53.85.252":["autodiscover.hlb.itglobalt.com","edge2016.hlb.itglobalt.com","mail.hlb.itglobalt.com"]}
```

En estos 4, solo el ultimo nos llegaría a interesar ya que es el único que tiene algo que ver con Repsol en una de sus IPs pero aun así, nos encontramos con un sub-dominio y no con un dominio por lo que el sistema autónomo REPSOL S.A. no nos sirve más asique empezaré a repetir el mismo proceso pero ahora con el siguiente siendo el Repsol Services Company.

```
┌──(kali㉿kali)-[~]
└─$ echo "\n" | curl https://sonar.omnisint.io/reverse/205.196.181.96/28
{"205.196.181.101":["smtp.talentforceinc.com"],"205.196.181.104":["ftpusa.repsol.com"],"205.196.181.107":["ushou-geoprx.repsol.com"]}

┌──(kali㉿kali)-[~]
└─$ echo "\n" | curl https://sonar.omnisint.io/reverse/209.173.253.96/27
{"209.173.253.109":["sip2.repsolsinopec.com","sip2.servexternos.repsolsinopec.com"],"209.173.253.112":["sip2.repsolsinopec.com","sip2.servexternos.repsolsinopec.com"],"209.173.253.116":["ushou-extranet.hlb.repsol.com"],"209.173.253.119":["rca.hlb.repsol.com"],"209.173.253.121":["wellview2.talisman-energy.com"],"209.173.253.122":["nimbus.repsol.com"]}
```

En este caso me salieron solo 7 rangos de IP y de los cuales solo 2 con resultados, pero volvemos al mismo problema de antes y es que son solo sub-dominios. Ya con esto nos quedaría ver el ultimo sistema autónomo, siendo este el Repsol Sinopec Resources UK Limited. En este caso pus no me saió ningún rango de red por lo que me quedo con los dos primeros de los cuales una información que puedo ir sacando son los subdominios que eran los siguientes:

- 195.53.85.250:vnpf5.repsol.com
- 205.196.181.104:ftpusa.repsol.com
- 205.196.181.107:ushou-geoprx.repsol.com
- 209.173.253.116:ushou-extranet.hlb.repsol.com
- 209.173.253.119: rca.hlb.repsol.com
- 209.173.253.122: nimbus.repsol.com

Aun siendo solo los su-dominios, nos da la suficiente información para entender que hay un dominio principal el cual es Repsol.com y que existe entre estas redes, y ahora veremos todos los subdominios que puede llegar a tener el Repsol.com

```
┌──(kali㉿kali)-[~]
└─$ curl https://sonar.omnisint.io/tlds/repsol
["repsol.com.ua","repsol.ro","repsol.store","repsol.training","repsol.kred","repsol.moscow","repsol.se","repsol.sk","repsol.com.br","repsol.gr","repsol.lv","repsol.be","repsol.club","repsol.do","repsol.market","repsol.pl","repsol.click","repsol.es","repsol.pt","repsol.cat","repsol.cz","repsol.earth","repsol.ru","repsol.ba","repsol.co","repsol.education","repsol.systems","repsol.uno","repsol.asia","repsol.com","repsol.com.pl","repsol.tech","repsol.cc","repsol.energy","repsol.no","repsol.site","repsol.xxx","repsol.hu","repsol.in","repsol.info","repsol.pe","repsol.in.ua","repsol.it","repsol.mx","repsol.org.ua","repsol.com.cn","repsol.fr","repsol.ir","repsol.mobi","repsol.ca","repsol.ch","repsol.com.mx","repsol.io","repsol.com.gr","repsol.tv","repsol.us","repsol.by","repsol.center","repsol.de","repsol.ec"]
```

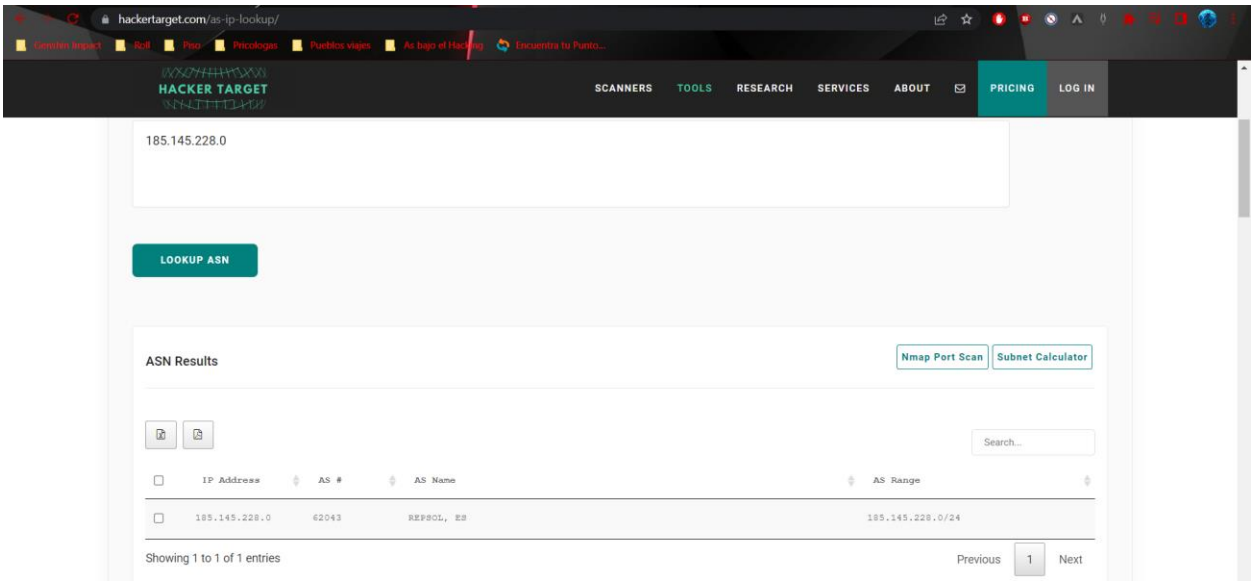
Adicionalmente pensé que, aunque no los encontráramos en los rangos de red, sería interesante mirar sobre otros dominios como es el Repsol.es y Repsol.ru aunque aparezcan muchos otros. Una vez empezando con esto, utilizaré la herramienta “Assetfinder” la cual nos permitirá encontrar/ver todos los sub-dominios (o casi todos) que haya de los dominios que he dicho anteriormente que iba a analizar. Viendo los resultados hablaré del que más me interesaba en mostrar que es en el de .com el cual pues quería buscar a ver si con el Assetfinder podría encontrar los que encontré con anterioridad a la hora de ver los rangos de IP.

```
┌──(kali㉿kali)-[~]
└─$ assetfinder repsol.com | grep ftpusa.repsol.com
ftpusa.repsol.com

┌──(kali㉿kali)-[~]
└─$ assetfinder repsol.com | grep vnpf5.repsol.com

┌──(kali㉿kali)-[~]
└─$ assetfinder repsol.com | grep ushou-geoprx.repsol.com                    1 ×
ushou-geoprx.repsol.com

┌──(kali㉿kali)-[~]
└─$ assetfinder repsol.com | grep ushou-extranet.hlb.repsol.com
ushou-extranet.hlb.repsol.com

┌──(kali㉿kali)-[~]
└─$ assetfinder repsol.com | grep rca.hlb.repsol.com
rca.hlb.repsol.com

┌──(kali㉿kali)-[~]
└─$ assetfinder repsol.com | grep nimbus.repsol.com
nimbus.repsol.com
nimbus.repsol.com
nimbus.repsol.com
nimbus.repsol.com
```

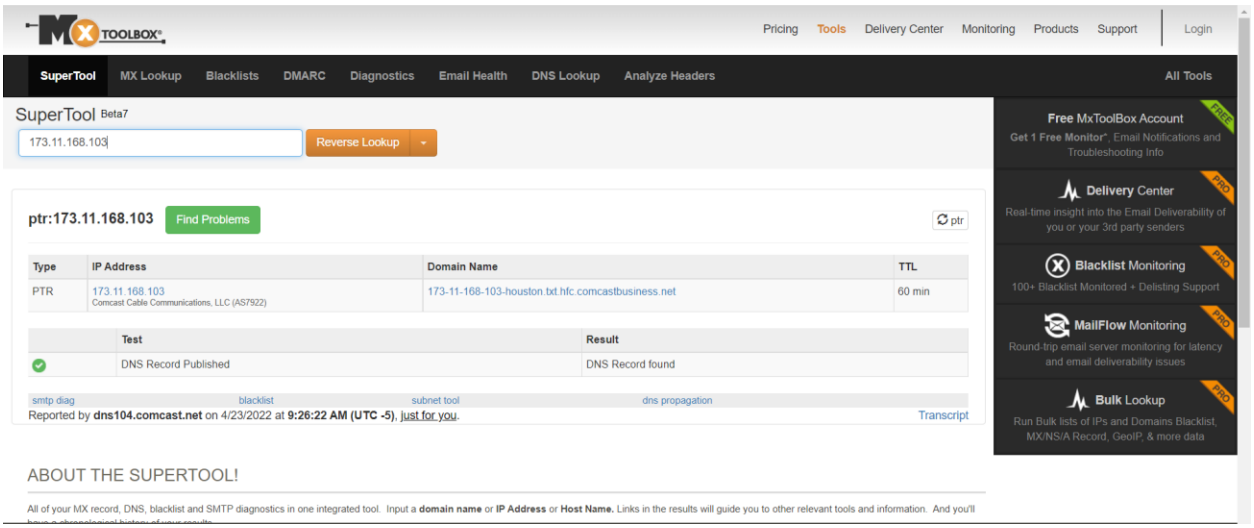
Como podemos ver si que fue posible hasta donde me sorprendió por un lado que no encontrásemos el de “vnpf5” pero sí que encontrásemos y varios de “nimbus” llegando hasta casi 20 resultados con este último. Pero en cualquier caso podemos ver que efectivamente estos subdominios existen y además gracias al análisis del CIDR que hicimos con sonar, pudimos hasta saber a qué IP exacta pertenecían donde ya sería cuestión de ver los servicios que presentan, buscar vectores y posibles vulnerabilidades y de que tratan en general, ya que si nos encontramos un caso donde solo guarda datos de quien viene al trabajo y quien no, u otros que no nos son de utilidad pues no nos servirían pero aun así, tenemos para elegir.

- Investigar técnicas/servicios alternativos (al menos 5) que permitan identificar activos que una empresa tenga expuestos en perímetro.

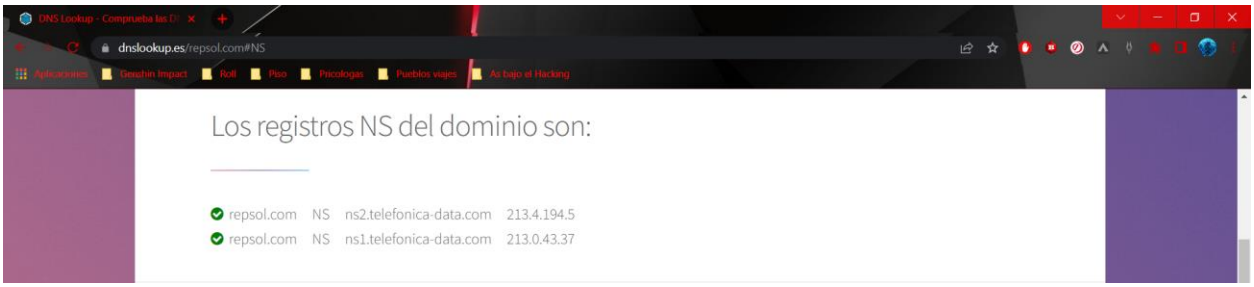
Para empezar, iré presentando de distintos ámbitos a la hora de identificar los activos como es como conseguir dominos, sub-dominios y etc. Y empezaré por ejemplo por un lado con Hacker Target el cual sirve para conseguir el nombre de los sistemas autónomos de las IPs que les pertenezca.



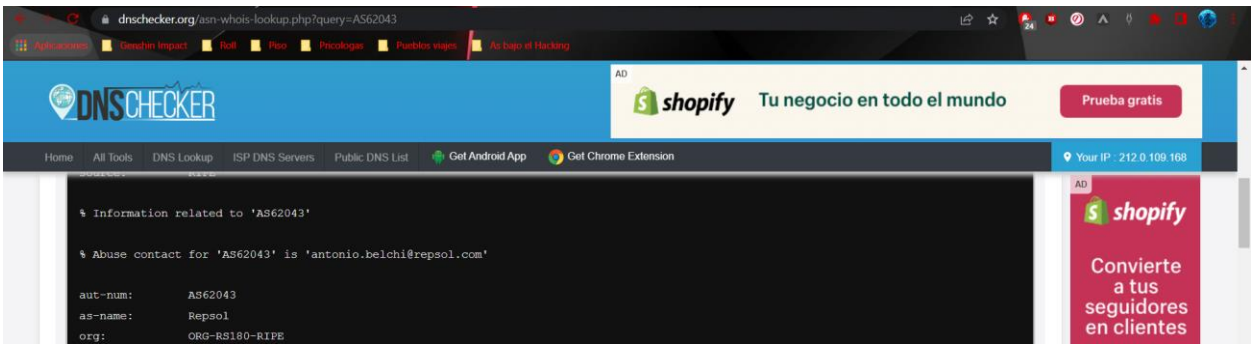
Por ejemplo, aquí lo que hago es poner una IP que se que pertenece a Repsol y donde pues al ponerla me sale que pertenece al sistema autónomo de Repsol y además de su rango, donde pues también se puede decir que esta página sirve también para sacar el rango de red.



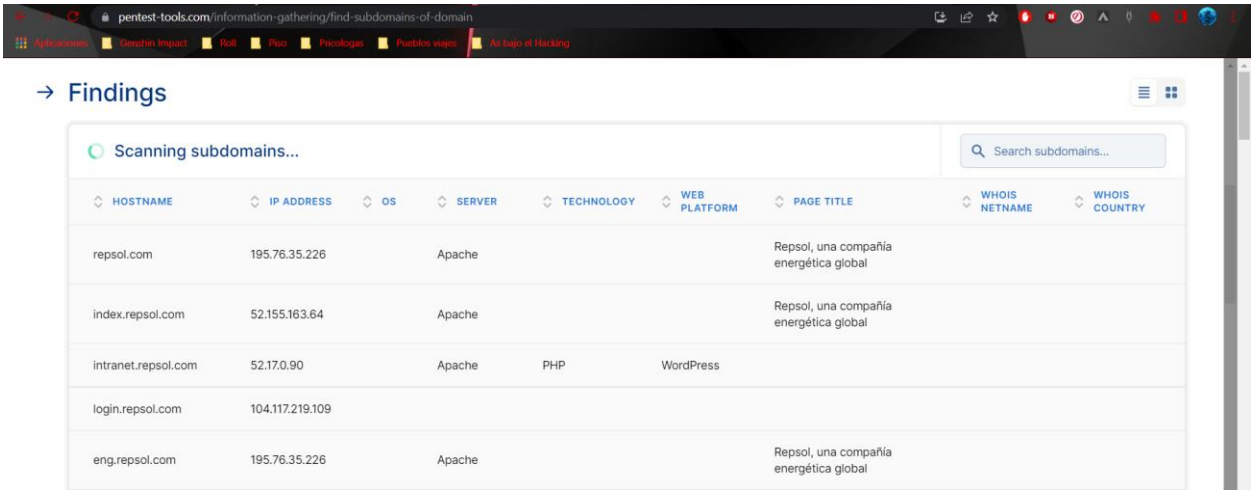
MX Toolbox es otra macro herramienta como es WhoisXMLapi donde te permite hacer casi todos los pasos de buscar dominios a subdominios, como probar el CIDR como sacar el rango de red si hace falta



Una tercera herramienta que existe es la de DndLoockup el cual nos permite realizar la técnica de Reverse Nslookup ya que nos da los NS de los dominios que le indicamos lo que ya con otra herramienta podríamos realizar dicha técnica.



Una herramienta más que nos ayuda con los sistemas autónomos sería el DNSCHECKAR el cual pues nos permite realizar distintas acciones, pero entre esta es mirar el número del sistema autónomos el cual si lo tuviésemos nos permitiría identificar a que sistema autónomo pertenece.

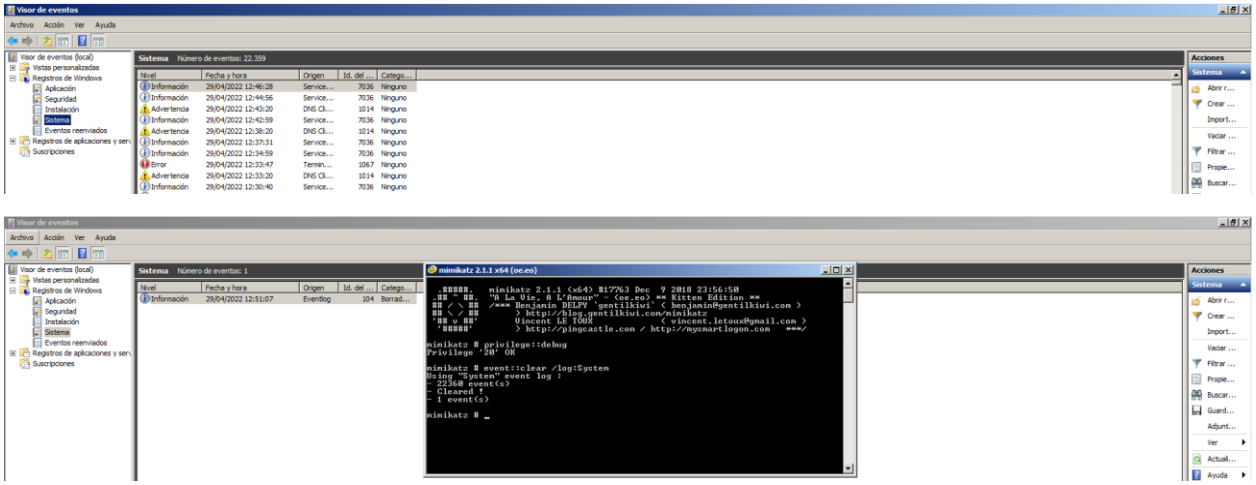


Como ultima herramienta sería una para localizar los subdominios de un dominio usando pentes-tools con los cuales luego también nos indican la IP especifica y luego a que servidor pertenece y luego otros datos que se puedan ver en la imagen.

- Investigar a fondo la herramienta Mimikatz, detallar y probar el funcionamiento de 5 técnicas diferentes que puedan desarrollarse con dicha herramienta y no hayan sido probadas en clase.

Mimikatz es una herramienta usada para muchísimas cosas, y conseguir contraseñas sería su uso por excelencia, pero aun así se puede usar para muchas otras funciones de las cuales explicaré y mostraré a continuación.

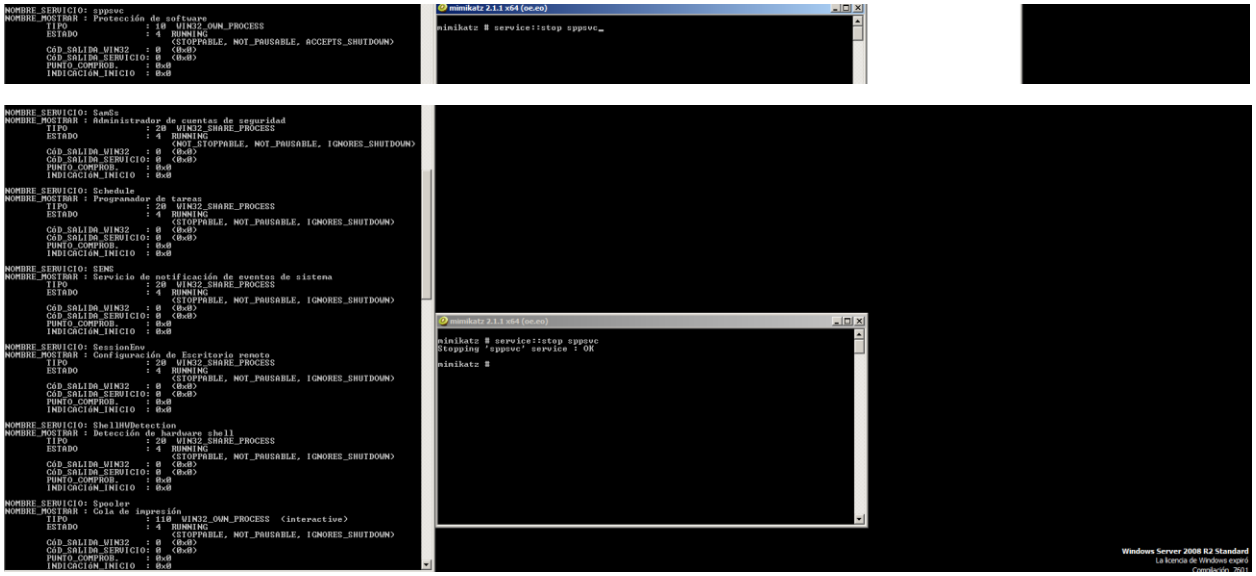
Event::clear

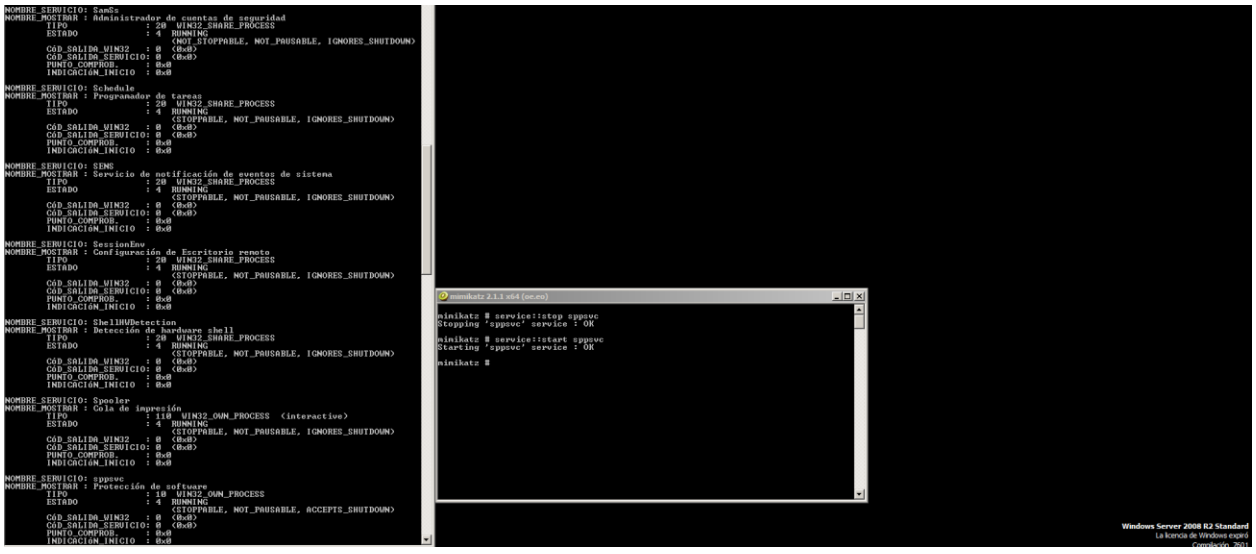


Event es un módulo que permite juguetear con los eventos de Windows de un ordenador como por ejemplo a editarlo, crear otros “nuevos” (realmente falsos) y por último el que voy a realizar yo que es el de borrar los eventos de un registro de Windows en concreto. Como se puede ver en la primera imagen, estoy en el visor de eventos de Windows y me meto a ver los registros que están en guardados y en el de “Sistema” para ser concreto donde pues podemos ver que está lleno de diferentes eventos. Ahora ya en

la segunda imagen se puede ver como con el mimikatz al lado lo que hago es por un lado ponerme privilegios y luego utilizo el módulo “evento::clear /log:System” haciendo que se borren todos los eventos del sistema, 22360 para ser exactos como nos indica el programa. Viendo como ha quedado el visor de eventos podemos observar que efectivamente se han borrado todos los eventos del sistema, a excepción de uno que realmente no es que no se haya borrado, sino que se ha creado con un evento que detalla que se han borrado los eventos. Como añadido podemos ver en mimikatz que hay un evento aun disponible y es el que se ha creado en el visor de eventos del Windows al haberse borrado todos estos, esto también nos sirve para limpiar nuestros rastros por ejemplo de una manera bastante eficiente.

Service::stop

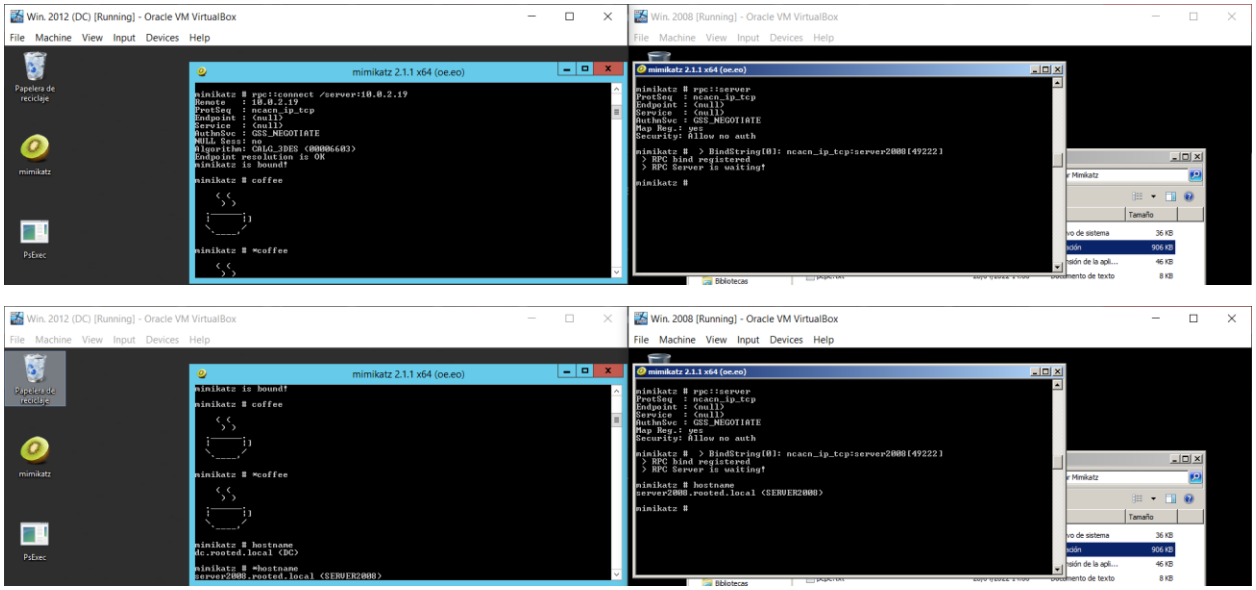




Service es un módulo potente el cual nos permite jugar con los distintos servicios del usuario como por ejemplo crear el mimikatzsvc en el equipo de la víctima a borrar o empezar distintos servicios, aunque es importante entender que no todos se pueden. Esto se debe a que hay algunos que sin su funcionamiento el sistema no funcionario correctamente, de la misma manera si eliminas la carpeta

“System” o “System32” de un ordenador el cual ya no es capaz de funcionar correctamente. Ahora bien, para el ejemplo a la izquierda de la imagen tengo un listando de todos los servicios activados del sistema con el comando “sc query” y luego a la derecha el mimikatz. Como se puede observar el servicio que he elegido ha sido un tanto especial ya que me fije en que tuviera dos etiquetas siendo estas las “STOPPABLE” y “ACCEPTS_SHUTDOWN” ya que si las tiene significa que puedo editar el servicio con el ejemplo que iba a enseñar que es el de parar los servicios. Como se puede ver en las dos siguientes imágenes, por un lado, cierro el servicio donde en el listado de servicios este desaparece y que como dije solo muestra los servicios habilitados y luego por otro lado en la tercera imagen lo habilito donde este vuelve a aparecer.

Rpc::connect



Este módulo se encarga de realizar conexiones entre dos ordenadores mediante una conexión RPC y para hacerlo, lo primero es tener pues instalado mimikatz en ambos sistemas, tanto del atacante como el de la víctima. Estando en un escenario perfecto donde hemos podido instalar mimikat n la víctima, lo que hacemos es activar el servicio de RPC con “rpc::server” donde al hacerlo pues nos aparecerá un mensaje con que se ha encendido el servicio correctamente. De ser así lo que haríamos pues el atacante simplemente tendría que escribir “rpc::connect /server:[IP victima]” para poder conectarse y saldrá un mensaje final de “mimikatz is bound!” donde confirma que se ha conectado correctamente. Una vez conectado ya podemos hacer distintas acciones y ver los resultados que se verían en el sistema de la víctima, pero desde nuestro ordenador y para ello hace falta solo poner un “*” antes del comando. Como podemos ver en el ejemplo en ela segunda imagen (ignorando lo del café) se podrá ver como si pongo “hostname” solo, pues aparece el hostname de nuestro ordenador, pero si lo pondo con el símbolo de antes, pues me pondrá del ordenador al cual estoy conectado y para demostrarlo podemos ver cómo me aparece este mismo en la otra maquina ya que es el suyo.

Crypto::providers

Con este módulo es bastante simple donde nos permite ver una lista de los proveedores criptográficos que tenga el sistema. Estos mismos tiene el objetivo de asegurar claves privadas de nuestro sistema y dependiendo de unos y otros, podremos ver si tiene algún tipo de fallo, pero como podemos ver en la imagen, cada proveedor indica la medida de seguridad de la clave junto a el nombre del proveedor además el mimikatz también nos lo divide por tipos.

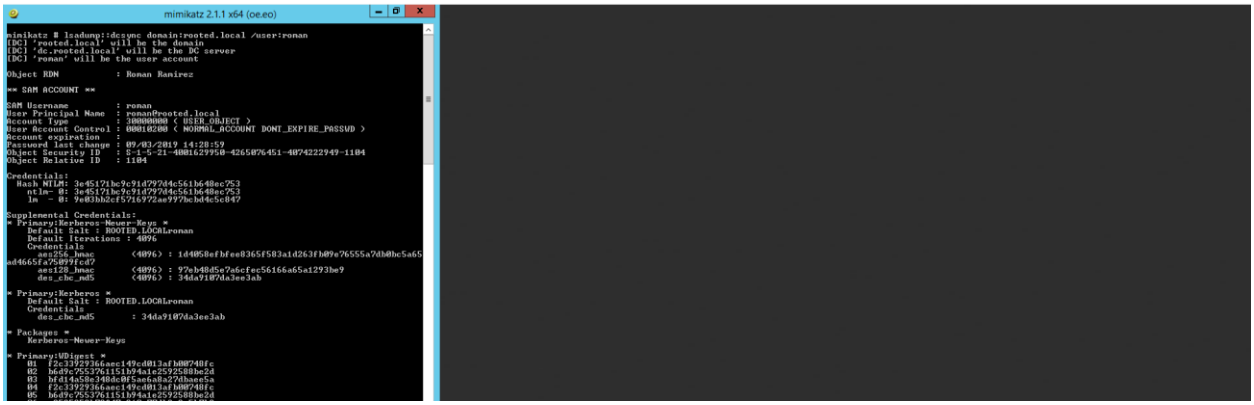
Crypto::hash

Esta variación del módulo de “Crypto” nos permite hashear distintas contraseñas u usuarios, ya se dónde hay claves cerradas y sabemos un hash, pues nos permitiría comprobar si una de las posibles contraseñas coincide con el hash, y este mismo tenemos hasta cinco tipos. Como podemos ver su funcionalidad es bastante simple, solo hay que poner “crypto::hash /password:[contraseña]” para que nos indique un hash de esta misma. Si nos fijáramos, introducir el usuario también nos genera otros dos hashes, siendo estos el DCC1 y DCC2 los cuales pasan por la cache de las credenciales del dominio (Domain Cache Credentials), estas mismas se utilizan para guardar la información de las claves de manera local para que el usuario pueda conectarse a un sistema si el servidor de inicio de sesión no estaba disponible. Este tipo de hash esta hasheado de tal manera de que estos tipos de hashes no sirven para la realización de la técnica de “pass-the-hash”. Como dato adicional estos mismos se pueden verse en diferentes sistemas como “MSCACHE” ya que es el algoritmo que en realidad se utiliza de fondo para crear el hash.

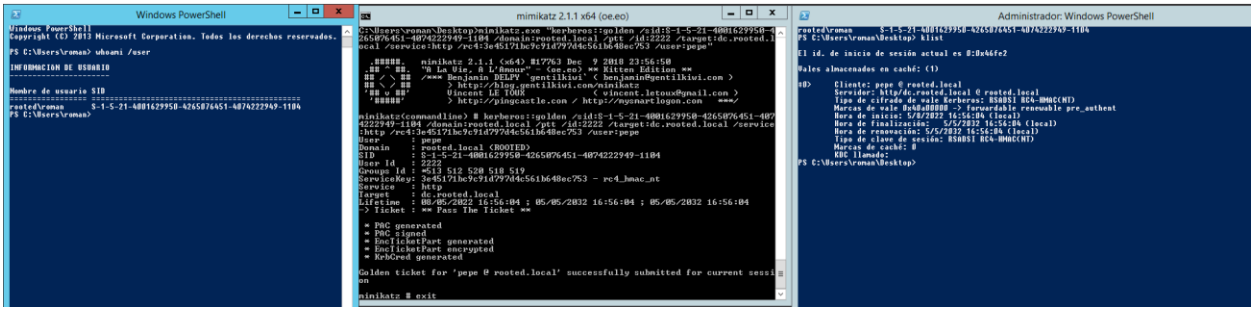
- Investigar en que consiste la técnica DCsync y otras dos de persistencia en dominio a elección del alumno, y comprobar su funcionamiento en el laboratorio proporcionado.

La técnica de DCsync consiste en hacerse pasar por un controlador de dominios el cual le pide a otro que le de información de contraseñas de diferentes usuarios y para realizar esto, hace uso del protocolo MS-DRSR (Directory Replication Service Remote Protocole). Lo bueno de esto es que se puede realizar este tipo de ataque sin tener que hacer ningún tipo de código sobre el control de domino víctima, pero para la simplificación de este sipo de ataque lo haré en este mismo y lo haremos sobre el usuario “roman” ya que

usaremos la información que nos dé para el ejemplo de las otras dos persistencias de dominio que presentaré.



Y así es como podemos sacar los datos de un usuario registrado en el DC como sus contraseñas, hashes y más información que nos sería útil, pero para lo que vamos a usar esto será para las otras dos funciones las cuales son el Silver Ticket y el Golden Ticket siendo esto dos persistencias en el dominio las cuales una es una versión mejorada de la otra, pero a su vez llama más atención. También una de las principales diferencias entre estas es que el Silver Ticket es un acceso para solo un servicio de un ordenador especifico mientras que el Golden Ticket es para cualquier servicio, pero como dije antes, ver a un usuario nuevo con el Golden Ticket es un poco extraño cuanto menos, sobre todo si viene sin ningún aviso de antelación mientras que un Silver Ticket se crea para todas las personas todo el rato ya que habrán sistemas que necesitan que acceder a otros ya que estos mismos tiene un tipo de servicio que necesitan. Primero realizaré el ejemplo de la primera, el Silver Ticket, este tipo de persistencia lo que hace es que un usuario (atacante) tenga cuando este quiera disponible poder acceder a un servicio de un sistema en específico y para esto lo que haremos será usa el Mimikatz como dije anteriormente.

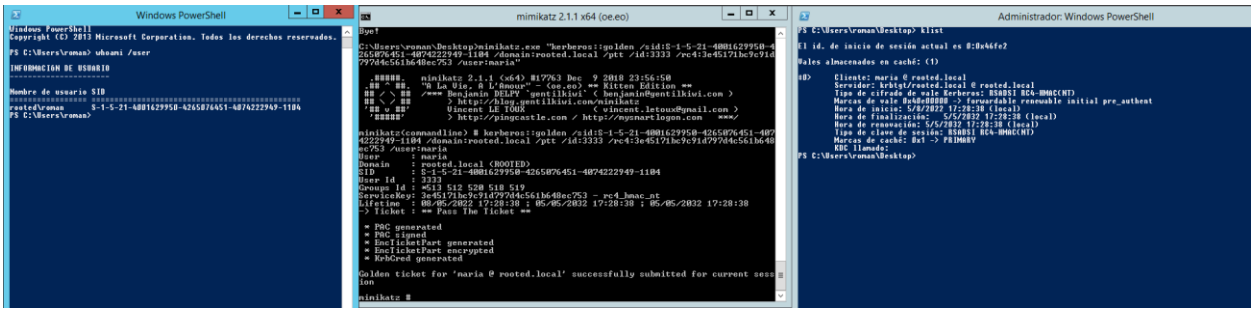


En este caso, el Silver Tiquet necesitas muchos parámetros de los cuales usamos son los siguientes:

- SID: El numero identificativo del usuario que nosotros usaremos para crear nuestro ticket
- Domain: El dominio en el cual vamos a crear el ticket
- Ptt: es un comando aparte el cual no haría falta, pero lo usamos para que nos meta directamente el ticket en memoria para que este pueda ser usado directamente.
- Id: Es un Id por el cual nuestro nuevo usuario pueda hacer uso del ticket que creemos.
- Target: El domain controller del cual vamos a sacar el ticket
- Service: el servicio para el cual creamos el ticket
- Rc4: Es el hash NTLM del usuario del cual sacamos en la parte del Dcsync

- User: Aquí simplemente ponemos el nombre del usuario a quien será asociado el tiquet el cual puede ser uno inventado

Una vez teniendo esto, podremos observar como en la imagen para conseguir el SID simplemente es el comando “whoami /user” y luego para los demás son información que podemos sacar usando “hostname” o metiéndonos en las propiedades del usuario. Pero, en cualquier caso, una vez tengamos esto y se nos ha generado el Silver Ticket (sé que se ve en la consola que pone que se generó el Golden Ticket, pero porque el método para realizar un Silver Ticket con Mimikatz es a través del Golden Ticket) el cual podemos ver que está en “klist” que es una lista en la cual podemos ver todos los Ticket que haya en el Domain Controller y a que usuario esta adjunto y etc.



Aquí ya hacemos el Golden Ticket en si con el cual como dije, podremos acceder a cualquier servicio de una maquina donde pues en este caso decidí crear a un usuario llamado Maria la cual pues es una usuaria la cual que tiene como dije acceso a cualquier servicio. Otro detalle así más que se puede ver es que a la hora de usar el mimikatz, podemos ver que hay dos parámetros que los quitamos como son el servicio (ya queda igual al poder conectarse a cualquier servicio y no uno en específico) y target (ya que es donde está el servicio guardado, y por la misma razón que el anterior, eso ya no importa ya que nos podemos conectar a cualquier servicio).Otro detalle que sería interesante comentar, es el hecho de que en la generación de los tickets es que hay un apartado que pone “Groups ID” el cual indica a que grupos pertenece el ticket que se creó, que en este caso estos representan: “Domain User”, “Domain Admin”, “Group Policy Creators Owners”, “Schema Admins” y “Enterprise Admin” respectivamente como va apareciendo en orden en la imagen.

- Investigar en que consiste la vulnerabilidad Zero-logon de Windows, indicando a que sistemas afecta, y evidenciando posteriormente su funcionamiento.

Zero-Logon es una vulnerabilidad que solo es aplicable a sistemas Windows desde los cuales, en resumen, es una vulnerabilidad que se encuentra en el inicio de sesión del protocolo MS-NRPC que pertenece a Windows Netlogon. El vector de inicialización es rellenado por ceros todo el tiempo además de que se le establece un numero aleatorio siendo esta una forma de hashear las claves privadas del sistema donde esto es aprovechado por los atacantes para hacerse pasar por cualquier sistema de una empresa, hasta teniendo la posibilidad de ser el controlador del dominio raíz pero no solo se queda ahí ya que teniendo

el control del Netlogon le permite al atacante tener un canal seguro de entre los diferentes sistemas de una empresa pudiendo así irse de un sistema a otro, plantar un exploit donde lo necesite llegando a ser hasta un entorno con el cual el atacante pueda jugar. EL atacante también podrá acceder a cualquier terminal o política de grupo y etc. con cualquier tipo de permiso que se necesite, como si es el más alto si hace falta donde pues, en resumen, este puede realizar cualquier acción desde donde y cuando quiera y como quiera.

Ahora para realizar el ataque de prueba, hay distintas herramientas como métodos para este mismo, pero en mi caso utilizaré la herramienta “imapket” que viene preinstalada en Kali, y sacaré también un programa de github que se llama “zerologon” para poder realizar el exploit en si.

Una vez lo tengo descargado, me meto dentro de la carptea para buscar un archivo Python que es el cual ejecutará el exploit en si.

Ahora como podemos ver, este ha funcionado y para si funcionamiento, lo que necesitábamos son el nombre del DC al cual íbamos a atacar el cual pues en este caso se llama “dc” y luego la IP de este mismo que para mi caso sería la 10.0.2.25. Con esta información, al ejecutar el ataque veremos el texto al final de la imagen en la cual nos dice que lo ha conseguido, y que ahora la contraseña del DC está vacía es decir, no tiene ningún carácter asique cuando necesitemos la contraseña, esta con darle al “ENTER” para avanzar será más que suficiente.

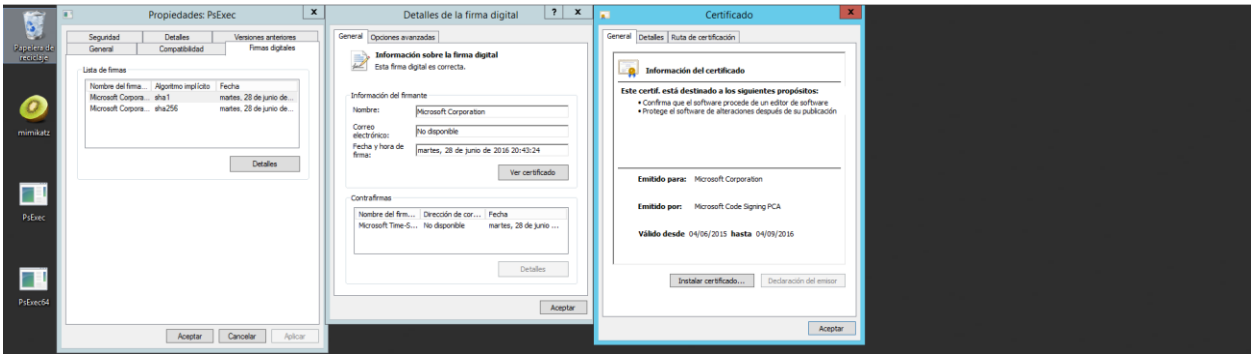
Como podemos ver, hemos usado ahora la herramienta “impacket” para este caso para que nos saque todos los hashes de cada usuario para podes pasar como estos y para ello hemos dicho el domain controller que en este caso es una maquina Windows 8 a la cual luego le indicamos la IP de esta misma. Luego me pide la contraseña para poder obtener dicha información, pero como ya no la necesitamos por la acción de antes pues al darle para que continue con la acción pues me sale todas las credenciales del dominio que le indiqué y entre estas, podemos ver algunas como de los diferentes usuarios como roman, jose y pepe, por ejemplo.

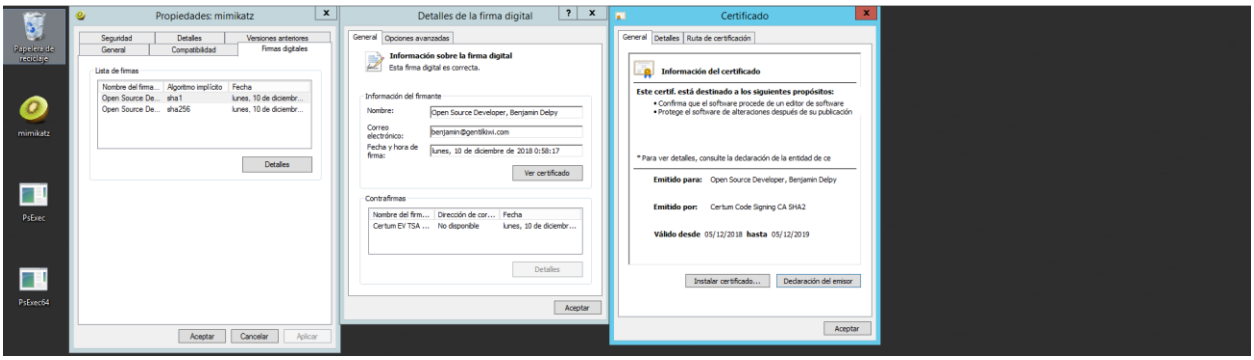
```
┌──(kali㉿kali)-[~/Documents/zerologon]
└─$ impacket-wmiexec rooted.local/Administrador@10.0.2.25 -hashes aad3b435b51404eeaad3b435b51404ee:930e8942be7879af7711f75e02b63cfb
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] SMBv3.0 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>whoami
rooted\administrador
```

De todas las credenciales de antes las que más me interesaban eran del administrador el cual pues para entrar como este lo que necesito es volver a usar la herramienta de imapcket pero esta vez otra versión de esta misma y para esta misma necesitaré pues indicar el usuario junto al dominio a los cuales luego se le añade su IP. Por último, le metemos el hash de antes que elegimos, que es el suyo para poder pasarnos como este mismo y al hacerlo nos abre una terminal con la cual podemos interactuar y para comprobar quien era le puse un “whoami” y efectivamente salí como admin.

- Investigar y demostrar como quitar firmas y lograr la ejecución de Mimikatz en un sistema con un antivirus a elección del alumno. Se espera que el binario/script sea detectado por menos de 5 AVs en VirusTotal.

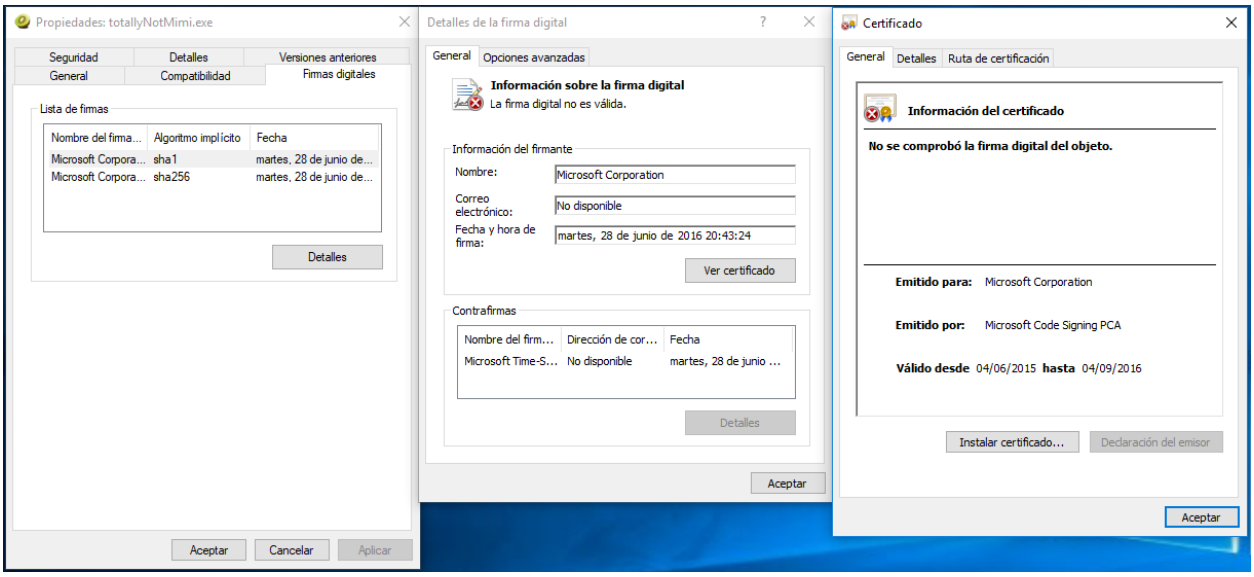
Par la acción del siguiente ejercicio, primero voy a meterme en la máquina virtual Windows server 2012 para mostrar los archivos que tenemos de los cuales indicaré de quien voy a quitar las firmas y porque





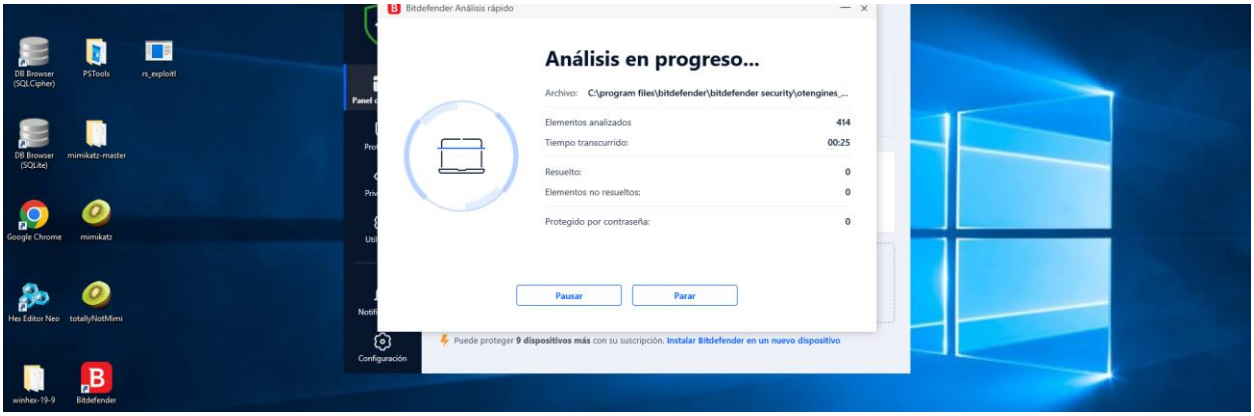
Como Podemos observer, el archivo elegido por mí ha sido el PsExec.exe ya que este mismo tiene la firma de Microsoft la cual pues permite al archivo estar en un sistema cual sea ya que al tener dicha forma pues

ningún antivirus debería de sospechar al ser un archivo que viene del propio Microsoft. Al tener esta firma confirma que el producto es de esta misma de manera que este podrá ejecutar comandos cuales sea que este diseñado para de manera que nada pueda interferir con este mismo, al contrario del Mimikatz donde pues a este mismo lo tiene visto muchos antiviruses y saben muy bien de donde es la firma del dicho programa. La idea de este tipo de actividad es quitarle la firma de manera que los antiviruses al fijarse en el programa, vean que se trata de un archivo con dicha firma y que por eso no deberían de tocarlo, aunque haga cosas a un nivel muy bajo de la computadora como es comprobar información confidencial de un equipo.

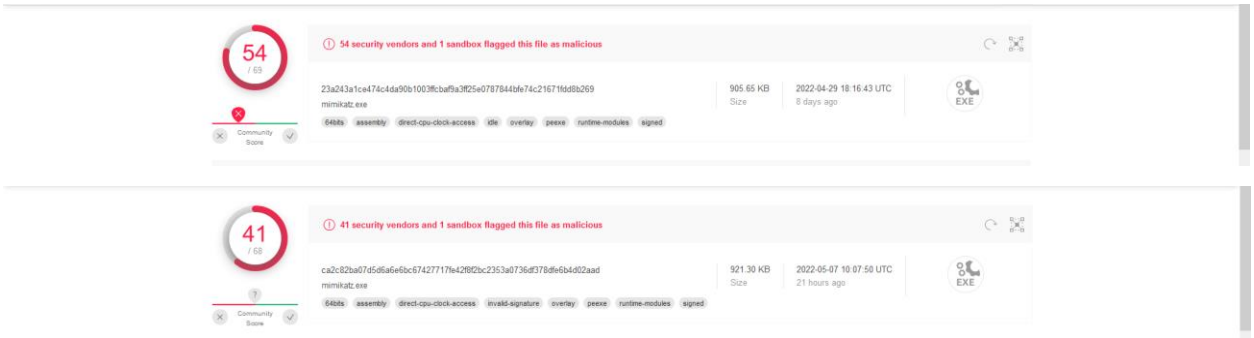


Para esto, he usado el “sigthief.py”, un archivo de Github donde lo que hace es esto mismo, coje a la firma de un archivo que le indiquemos y que se la pone a otro mismo, cual es el problema de esto mismo, y es que no puede replicar por desgracia la firma de este mismo perfectamente y por ende este mismo da error. También probé otras herramientas como son la de SigPirate.exe o metatwin pero ninguna me dio resultados por lo que los terminé dejando.

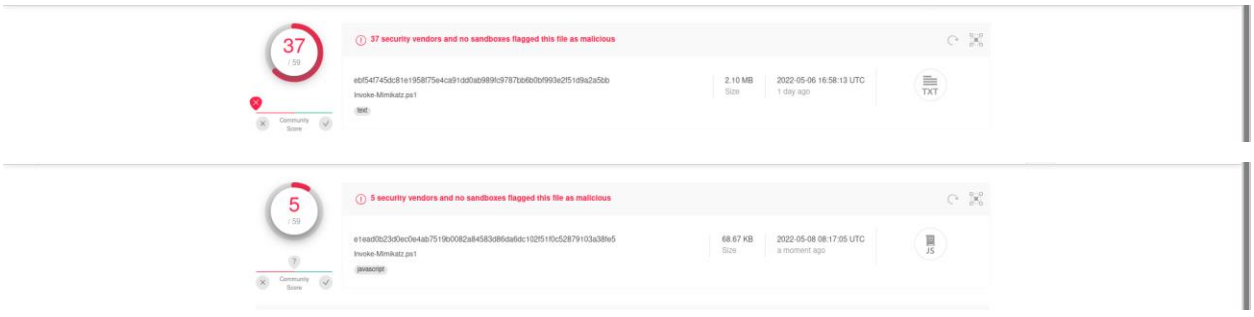
Por otro lado, una cosa que sí que probé fue pasar el virus de un lado de la maquino a otro y a ver si me lo cortaba el firewall del Windows Defender y a mi sorpresa no lo hacía.



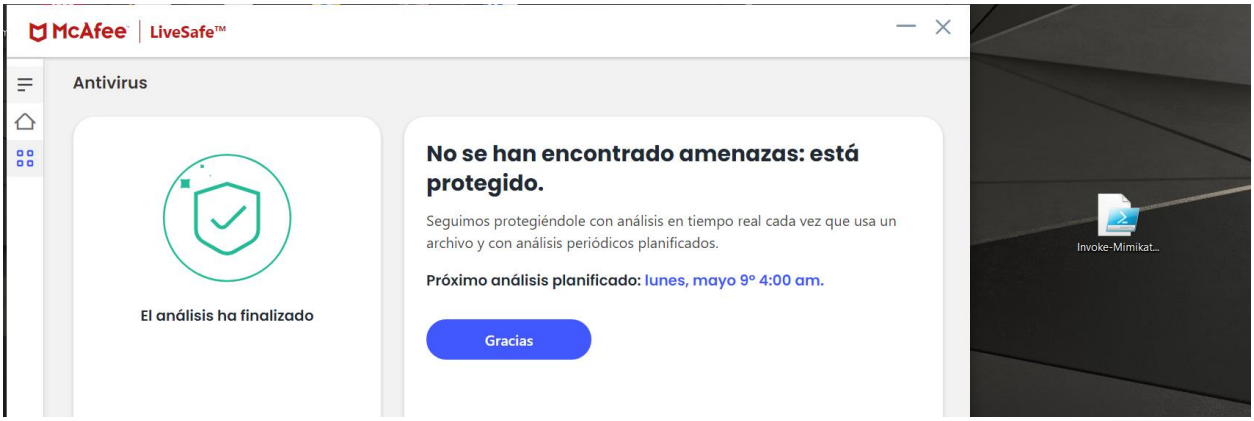
Por otro lado, me decidí descargar y meterme la suscripción gratuita de un mes del BitDefender y ver si me encontraba el archivo y pues como era de esperar pues sí que lo encontraba, aunque me dejase ejecutarlo, pero porque le dé indicaba que lo hiciese porque si no me lo paraba. Y lo mismo pasaba con el McAfee donde este sí que pago anualmente una suscripción y el cual ni si quiera me deja instalarlo y la única manera que me dejaba era usando la función de compartir archivos de VirtualBox entre los equipos, pero, aun así, me los borraba al poco rato.

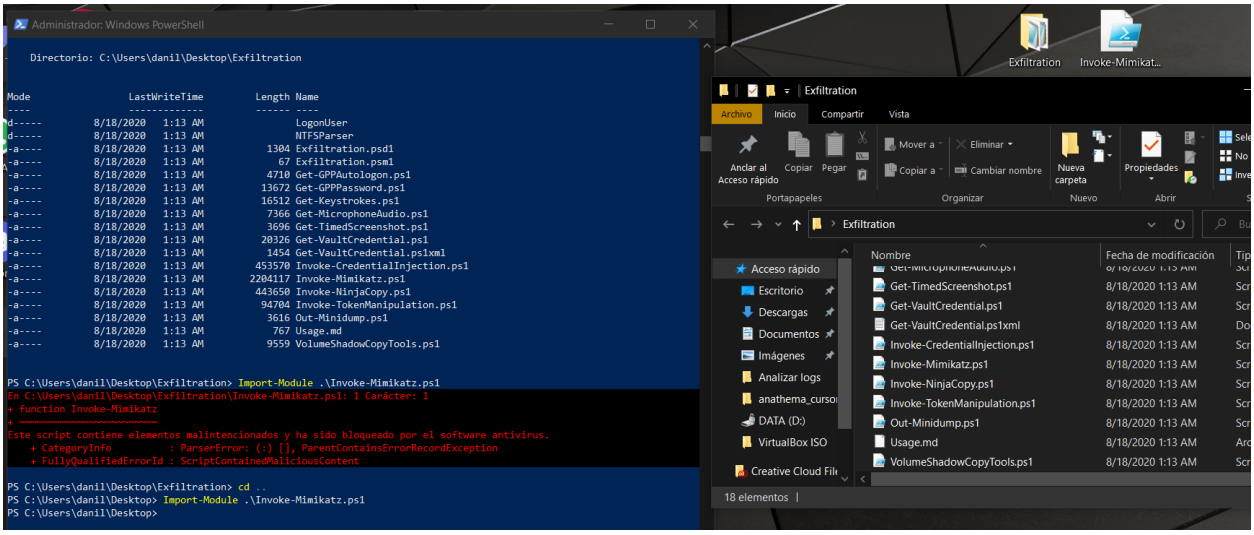


En estas imágenes podemos ver como lo que hice fue meter en Virus total los dos archivos mimikatz que hice, donde en el primer caso es el mimikatz normal sin ninguna modificación, mientras que el segundo es el con el cual le cambié las firmas. Como se puede ver, sí que hay un cambio semi considerable en el cual pues vemos como hay 13 antiviruses menos de los cuales no pillan el mimikatz con las firmas no autorizadas de Microsoft, y supongo de haber una herramienta que lo hiciese mejor podría llegar a más resultado y todo.



también quisiera presentar otra forma de realizar esto mismo y es de la siguiente manera, Invoke-Mimikatz.ps1 es un script de PowerShell que permite realizar las funciones de este mismo, pero desde PowerShell. Esto de por si es un cambio bastante grande ya que es más difícil de ver si un PS es un programa maligno o no, pero en cualquier caso en la primera imagen es este sin ningún cambio, es decir el original donde ya por sí solo hace diferencia con los otros dos. La diferencia de este mismo es de 17 antiviruses comparado al mimikatz original, pero si también le pasamos una herramienta para ocultar este mismo, la cual pues es Invoke-Stealth donde nos permite ocultarlo aun más consiguiendo el resultado que pedía el ejercicio el cual era que menos de 5 antiviruses pudieran pillar este mismo. El problema de este mismo es que también se especifica el ejercicio que tiene que sé por firmas este mismo donde pues esto sería una versión alternativa de poder conseguir esta parte del ejercicio, pero una cosa que ayuda a que lo encuentren muchos menos antiviruses es que no requiere de firmas, al ser un archivo .ps1.





Tan bueno es que hasta realizando pruebas en mi propio equipo ya que ahí es donde tengo el antivirus, al intentar realizar una búsqueda por todo mi ordenador, el McAfee no me lo encuentra el archivo y aun buscándolo por todo ese mismo. De la misma manera para ver al diferencia, me descargue los archivos PowerShell para realizar diferentes ataques maliciosos y ver que es lo que hacía cuando intentaba importarle el módulo original comparado al modificado pues el original me lo detectaba como se puede ver en la segunda imagen (y me lo borraba por lo que tenía que ejecutar esto y volver a instalarlo para poder mostrar el original en la imagen) mientras que en el modificado sí que me dejaba interactuar con este mismo como si fuese un archivo .ps1 normal y corriente.

Bibliografía:

- JoelGMSec, (Octubre de 2019), Invoke-Stealth, https://github.com/JoelGMSec/Invoke-Stealth
- Secretsquirrel, (Agosto de 2021), SigThief, https://github.com/secretsquirrel/SigThief
- B. Johansson, (Sin Fecha), Las 5 mejores protecciones antivirus gratis para Linux – 2022, https://es.safetydetectives.com/blog/mejores-antivirus-realmente-gratis-para-linux/
- Programador clic, (Sin Fecha), powershell + Invoke-Mimikatz.ps1 para obtener la contraseña del sistema,https://programmerclick.com/article/25991565417/
- Forsenergy, (Sin Fecha), Invoke-Expression, https://forsenergy.com/es-es/windowspowershellhelp/html/04b8e90a-7d28-4ab2-ad13-b0316c231c77.htm
- HackTricks, (Sin Fecha), Stealing Credentials, https://book.hacktricks.xyz/windows-hardening/stealing-credentials
- PowerShellMafia, (Agosto de 2020), PowerSploit, https://github.com/PowerShellMafia/PowerSploit
- J. Warren, (Julio de 2017), EXTRACTING USER PASSWORD DATA WITH MIMIKATZ DCSYNC, https://stealthbits.com/blog/extracting-user-password-data-with-mimikatz-dcsync/
- Red Teaming Experiments, (Sin Fecha), Kerberos: Silver Tickets https://www.ired.team/offensive-security-experiments/active-directory-kerberos-abuse/kerberos-silver-tickets
- Red Teaming Experiments, (Sin Fecha), Kerberos: Golden Tickets, https://www.ired.team/offensive-security-experiments/active-directory-kerberos-abuse/kerberos-golden-tickets
- Administrator, (Noviembre de 2017), Hijacking Digital Signatures, https://pentestlab.blog/2017/11/06/hijacking-digital-signatures/
- Threatexpress, (Octubre de 2017),metatwin, https://github.com/threatexpress/metatwin
- Imaibou, (Sin Fecha), mimikatz_obfuscator.sh, https://gist.github.com/imaibou/92feba3455bf173f123fbe50bbe80781
- C. Roberts, (Enero de 2017), How to Bypass Anti-Virus to Run Mimikatz, https://www.blackhillsinfosec.com/bypass-anti-virus-run-mimikatz/
- Icy Science, (2022), ¿Qué es un vector de inicialización? - definición de techopedia - Seguridad – 2022, https://es.theastrologypage.com/initialization-vector
- The Hacker Recipes, (Sin Fecha), ZeroLogon, https://www.thehacker.recipes/ad/movement/netlogon/zerologon
- The Hacker Recipes, (Sin Fecha), clear, https://tools.thehacker.recipes/mimikatz/modules/event/clear
- The Hacker Recipes, (Sin Fecha), hash, https://tools.thehacker.recipes/mimikatz/modules/crypto/hash
- The Hacker Recipes, (Sin Fecha), providers, https://tools.thehacker.recipes/mimikatz/modules/crypto/providers
- The Hacker Recipes, (Sin Fecha), Modules, https://tools.thehacker.recipes/mimikatz/modules#ts
- The Hacker Recipes, (Sin Fecha), +, https://tools.thehacker.recipes/mimikatz/modules/service/+
- Ayudaley, (Sin Fecha), Zerologon. Una vulnerabilidad de Windows Server muy peligrosa, https://ayudaleyprotecciondatos.es/2021/10/14/zerologon/#:~:text=Zerologon%20es%20una%20vulnerabilidad%20cr%C3%ADtica,el%20controlador%20de%20dominio%20ra%C3%ADz.