

Reconocimiento de Información

Caso práctico

Gerard Díaz Hoyos

KeepCoding
Bootcamp de Ciberseguridad
5a edición













En el presente documento voy a desarrollar la práctica del módulo de Reconocimiento de Información.

Para ello he escogido el website de **Magic Eden**, un marketplace de NFTs.



Observando los bug bounties de la web *hackerone.com*, vemos que **Magic Eden** se postula como posible plataforma online para ser analizada y “atacada” en los límites que ellos establecen, con ánimo de mejorar la ciberseguridad aplicada en su web.

El scope mostrado es el siguiente:

In Scope			
Domain	magiceden.io	 Critical	 Eligible
Domain	*.magiceden.io	 Critical	 Eligible
Domain	*.magiceden.dev	 Critical	 Eligible
Domain	coralcube.io	 Critical	 Eligible
Domain	*.coralcube.io	 Critical	 Eligible
Domain	*.magiceden.workers.dev	 Critical	 Eligible

Centraré los análisis posteriores en los siguientes dominios:

- 1) magiceden.io
- 2) magiceden.dev
- 3) coralcube.io

FOOTPRINTING:

Inicio el análisis realizando un reconocimiento vertical mediante la herramienta **Amass**, que realiza técnicas de reconocimiento pasivo y activo, de OSINT, etc.

Antes de comenzar a usarla, se configura el archivo *config.ini* con todas las variables y parámetros que se quieran aplicar en el momento de lanzar la herramienta (adjunto el archivo dentro de la carpeta que contendrá esta práctica). De este modo, indicamos los dominios que queremos analizar para encontrar todos los subdominios posibles, desactivamos la fuerza bruta (que la realizaremos posteriormente), desactivamos también las permutaciones por el mismo motivo, añadimos alguna *API key* de servicios de los cuales disponemos de cuenta (en mi caso sólo he puesto la de **Github** y **Censys**), etc.

Y a continuación, lanzaremos **Amass** a través de la consola de Linux (desde la máquina virtual Kali).

```
$ amass enum -src -v -config config.ini
```

En este caso en concreto se obtienen 52 resultados:

Como me interesa tener en un listado sólo los subdominios obtenidos, procedo a lanzar el siguiente script para dejar en el archivo.txt obtenido el contenido que me interesa:

```
$ cat amass.txt | awk '{print $2}' > subdomains-amass
```

Obtenemos el archivo subdomains-amass

A continuación realizaré fuerza bruta para seguir intentando sacar el máximo de subdominios posible.

Aunque, primeramente, utilizaré la herramienta **DNSValidator** para sacar un listado de DNS válidas con el cual realizar la fuerza bruta al dominio raíz con algún diccionario de los que tenemos publicados en Internet (*SecLists*).

```
-$ dnsvalidator -tL https://public-dns.info/nameservers.txt -threads 30 -o dsnvalidator-resolvers.txt
```

De aquí sacamos un documento tipo *.txt* con 824 registros validados llamado *dsnvalidator-resolvers.txt*.

Ahora pasamos a utilizar la herramienta **PureDNS** para realizar fuerza bruta al dominio principal de [magiceden.io](#). Haremos 2 pasadas para utilizar 2 diccionarios diferentes.

1a pasada:

```
└─$ puredns bruteforce /home/kali/Tools_cyber/SecLists/Discovery/DNS/namelist.txt magiceden.io -r dsnavalidator-resolvers.tx
t -w MagicEden-bruteforce01
```

Generamos el documento *MagicEden-bruteforce01* con 15 subdominios válidos.

2a pasada:

```
└─$ puredns bruteforce /home/kali/Tools_cyber/SecLists/Discovery/DNS/shubs-subdomains.txt magiceden.io -r dsnsvalidator-reso  
lvers.txt -w MagicEden-bruteforce02
```

Generando, esta vez, el documento *MagicEden-bruteforce02* con subdominios válidos.

Y haremos una 3a pasada al listado de subdominos obtenidos lanzando amass anteriormente:

```
└─$ puredns bruteforce /home/kali/Tools_cyber/SecLists/Discovery/DNS/namelist.txt subdomains-amass -r dsnvalidator-resolver.s.txt -w MagicEden-bruteforce-amass01
```

Pero en este caso no consigo ningún resultado, pese a haberlo probado en varias ocasiones y descendiendo la velocidad de procesado o análisis.

```
Resolving domains with public resolvers  
[ETA 00:00:00] | ████████████████████████████████████████ | 151265/151265 rate: 11311 qps (time: 00:00:12)  
  
Detecting wildcard root subdomains  
[ETA 00:00:00] | ████████████████████████████████████████ | 306/306 queries: 0 (time: 00:00:00)  
  
Validating domains against trusted resolvers  
[ETA 00:00:00] | ████████████████████████████████████████ | 306/306 rate: 306 qps (time: 00:00:00)  
  
No valid domains remaining.
```

Junto todos los resultados, filtrando para que no se repitan con el siguiente script:

```
$ sort -u MagicEden-bruteforce01 MagicEden-bruteforce02 subdomains-amass > subdominios-MagicEden
```

A continuación, procedo a realizar las permutaciones correspondientes con la herramienta **Gotator**, utilizando uno de los diccionarios de prefijos que aparecen en las SecLists.

```
$ gotator -sub subdominios-MagicEden -perm /home/kali/Tools_cyber/SecLists/Discovery/DNS/deepmagic.com-prefixes-top500.txt -depth 1 -numbers 10 -mindup -adv -md > gotator01.txt
```

En este caso obtenemos 422.519 registros.

Vamos a resolver esta gran cantidad de registros con **Puredns** resolve:

```
$ puredns resolve gotator01.txt -r dsnvalidator-resolvers.txt -w subdominios-MagicEden1
```

Obteniendo 65 subdominios validados.

Ahora, con ánimo de seguir filtrando y obteniendo más subdominios, vamos a analizar los certificados mediante la herramienta **Cero**:

```
$ cero < subdominios-MagicEden1 | grep magiceden.io > cero.txt
```

y después de filtrar:

```
$ sort -u cero.txt > cerol.txt
```

obtenemos 20 subdominios más.

A partir de aquí ya se han reunido suficientes subdominios con la aplicación de las técnicas anteriores, teniendo ya el archivo **subdominios-final.txt**.

Utilizo la herramienta **DNSx** para resolver estos subdominios y obtener el máximo número de IP's válidas:

```
$ dnsx -a -resp-only -l subdominios-final.txt -o ips-final.txt
```

...que tras filtrar para eliminar los registros idénticos, obtengo 29 IP's.

Obtengo, así, el archivo **ips-final-unique.txt**.

Y para preparar la fase de análisis de vulnerabilidades, utilizo una vez más **DNSx** para encontrar aquellos subdominios que son *nombres canónicos* -**CNAME**-, es decir, aquellos que están apuntando a otro subdominio en vez de a una IP. Esta información es relevante porque estos subdominios son susceptibles de recibir el ataque conocido como **Subdomain Takeover**, que permite a los atacantes hacerse con el control total del mismo, subir archivos, etc.

```
$ dnsx -cname -resp-only -l subdominios-final.txt -o cname-final.txt
```

Localizo 8 subdominios CNAME y generamos el archivo **cname-final.txt**.

Con los documentos **subdominios-final.txt**, **ips-final-unique.txt** y **cname-final.txt** ya puedo iniciar la siguiente fase del reconocimiento de información.

FINGERPRINTING:

Para comenzar esta etapa de identificación, se deberían escanear los puertos de las direcciones IP's que se han ido encontrando en los pasos anteriores (en concreto, 65.535 por cada IP).

Debido a la gran cantidad de puertos, se suele utilizar para dar una primera pasada la herramienta **Masscan**, que como su propio nombre indica, está pensada para otorgar velocidad “masiva” al escaneo y, por ello, pensada para el escaneo de grandes números de puertos.

En mi caso, tras intentarlo 3 veces, no he podido obtener resultados válidos, puesto que o me daba 0 puertos encontrados o tan sólo 2 si bajaba la velocidad de escaneo.

```
$ sudo masscan -p0- -iL ips-final-unique.txt --rate=5000 -oX masscan.xml
```

```
$ sudo masscan -p0- -iL ips-final-unique.txt --rate=1000 -oX masscan.xml
```

Not working...

Debido a que el tiempo que requería para completar el análisis era demasiado elevado, he optado por realizar los escaneos directamente con **Nmap**.

```
$ nmap -T5 --top-ports 5000 -iL ips-final-unique.txt
```

Obteniendo así el archivo nmap-ips-magiceden.txt

Debido a que no me funciona **Masscan**, **httpx**, **Wafw00f**, debo dejarlo aquí y continuaré con ella en la siguiente entrega...

Ha quedado pendiente el análisis de vulnerabilidades que lo hubiera hecho con la herramienta **Nicto**.

Y la parte de **OSINT**, que no me ha dado tiempo a repasarlo adecuadamente.