Seguridad en Redes de Comunicación

Práctica 3. EAP – RADIUS – IEEE 802.1X

Jorge Lanza Calderón

Luis Sánchez González

Departamento de Ingeniería de Comunicaciones

PRÁCTICA 3
EAP – RADIUS – IEEE 802.1X

1. Objetivos de la Práctica

El desarrollo de la práctica pretende complementar el aprendizaje de los conceptos teóricos que se han visto durante las sesiones de teoría.

Los principales objetivos de esta práctica son:

- Analizar y comprender los mecanismos y protocolos involucrados en la autenticación basada en EAP, RADIUS e IEEE 802.1X.
- Configurar y emplear el servidor de código libre para Linux FreeRADIUS.
- Monitorizar y analizar distintos métodos de autenticación EAP.
- Analizar el acceso a una red IEEE 802.11 protegida mediante WPA2-Enterprise.

2. Introducción

Como hemos visto en clase, EAP es un protocolo que define un marco genérico para autenticación que usualmente no se utiliza de manera independiente sino embebido dentro de otros protocolos.

A lo largo de la práctica analizaremos la forma en la que interactúan dispositivos para autenticarse utilizando una combinación de los protocolos EAP, RADIUS e IEEE 802.1X.

A la conclusión de la práctica experimentaremos con el acceso a redes IEEE 802.11 basado en IEEE 802.1X para monitorizar el empleo de estos protocolos en situaciones cotidianas.

3. Desarrollo de la Práctica

La práctica se desarrollará en un entorno Linux, para lo cual los datos de acceso a la cuenta definidos son:

```
Nombre de usuario: Alumnos General
Contraseña: telematica
```

3.1 Configuración del servidor RADIUS

Dentro de la arquitectura de red que hemos visto en las sesiones de teoría, el elemento que realizaba las funciones de Autenticación y Autorización era el servidor RADIUS.

FreeRADIUS (www.freeradius.org) es una de las múltiples implementaciones de servidores RADIUS para Linux.

Las posibilidades de configuración de este servidor son muy grandes, pero en esta práctica sólo nos concentraremos en su operativa básica.

Para ello, emplearemos tres archivos de configuración que tenéis en `/home/alumnos/Asignaturas/SRC/eap`:

- users
- clients.conf
- eap.conf

El primero de ellos contendrá la información relativa a los usuarios del sistema. En este sentido, tendréis que **editar este fichero e incluir en él una línea que contenga el nombre de usuario, y su correspondiente password, que utilizaréis a lo largo de la práctica**.

La sintaxis para añadir usuarios es la siguiente:

```
<nombre de usuario> Cleartext-Password := "<password>"
```

En la arquitectura de red de RADIUS, el papel de cliente lo asume normalmente el elemento que da acceso a la red (punto de acceso WiFi, switch Ethernet, etc.) sin embargo para el desarrollo de la práctica, usaremos una aplicación software que ejecutaremos desde la propia máquina donde estará ejecutándose el servidor RADIUS. El archivo `clients.conf` contiene la información de configuración relativa a los clientes que se pueden conectar con el servidor.

El contenido del archivo es el siguiente:

```
client localhost {
      ipaddr = 127.0.0.1
      secret = <secreto_cliente_servidor>
      require_message_authenticator = no
      nastype = other
}
```

Tendréis que **editar el archivo y elegir cual es el secreto que queréis emplear para proteger la comunicación entre cliente y servidor RADIUS**.

Por último, el archivo `eap.conf` contiene las opciones de configuración que definen la autenticación RADIUS basada en métodos EAP. Modificaremos este archivo a lo largo de la práctica dependiendo del método de autenticación que vayamos a usar en cada momento.

El contenido de este archivo es

```
eap {
      # Invoke the default supported EAP type when
      # EAP-Identity response is received.
      default_eap_type = md5
      #default_eap_type = ttls
      #default_eap_type = tls
      #default_eap_type = peap

      timer_expire     = 60
      ignore_unknown_eap_types = no
      max_sessions = 4096

      # Supported EAP-types

      # EAP-MD5
      md5 {
      }

      # MS-CHAPv2
      mschapv2 {
      }
}
```

El parámetro `default_eap_type` define cuál es el método EAP preferido por el servidor RADIUS. Inicialmente está fijado al uso de EAP-MD5 aunque aparecen comentadas otras opciones como EAP-TTLS, EAP-TLS o PEAP.

Por el momento no es necesario modificar este archivo.

Para lanzar el servidor RADIUS una vez configurado es necesario ejecutar el siguiente comando:

```
$ sudo freeradius -X
```

Comprobad que el servidor está corriendo correctamente asegurando que en la terminal aparece un mensaje similar al siguiente:

```
Listening on authentication address * port 1812
Listening on authentication address 127.0.0.1 port 18120 as server inner-tunnel
Listening on proxy address * port 1814
Ready to process requests.
```

El servidor bloquea esa terminal y todos los mensajes de log del servidor RADIUS se volcarán sobre ella.

3.2 Autenticación basada en EAP-MD5

Una vez que el servidor está corriendo generaremos procesos de autenticación basados en diferentes métodos EAP.

Para ello, utilizaremos la aplicación `rad_eap_test` que encontraréis en `/home/alumnos/Asignaturas/SRC/rad_eap_test-0.23`.

Ejecutando:

```
$ ./rad_eap_test
```

Podréis accede a una suerte de manual (también disponible en http://www.eduroam.cz/rad_eap_test/rad_eap_test.html)

Debéis **ejecutar el comando que os permita generar un proceso de autenticación basado en EAP-MD5**.

Las opciones del comando que debéis utilizar son:

-H *<address>*
    Address of radius server
-P *<port>*
    Port of radius server
-S *<secret>*
    Secret for radius server communication
-u *<username>*
    Username
-p *<password>*
    Password
-e *<method>*
    EAP method (PEAP | TLS | TTLS | LEAP | MD5)
-m *<method>*
    IEEE8021X

Capturad y analizad el intercambio de paquetes usando Wireshark.

Probad lo que ocurre con la autenticación si se introduce un usuario que no aparece en el archivo `users` o no se introduce correctamente el password.

Comprobad y verificad el correcto funcionamiento del EAP-MD5 Challenge / Response. Para ello, obtened el Challenge MD5 de la captura de Wireshark, realizad manualmente los cálculos para obtener el correspondiente Response y comparad éste con el campo correspondiente en la captura.

Para el cálculo manual del Response recordad:

```
xxd - make a hexdump or do the reverse.

$ echo <ASCII string> | xxd
```

Transformaría el string ASCII que le pasemos a su correspondiente dump de bytes en formato hexadecimal.

```
$ xxd -r -p file.txt file.bin
```

Transformaría el dump de bytes en formato hexadecimal contenido en *file.txt* en el valor de estos bytes en formato binario y lo almacenaría en *file.bin*.

```
md5sum - compute and check MD5 message digest

$ md5sum file.bin
```

generaría el valor del hash MD5 de los bytes contenidos en el fichero *file.bin*.

Comprobad y verificad el correcto funcionamiento de alguno de los RADIUS Authenticator Request / Response. Para ello, obtened el valor del campo Authenticator de alguno de los mensajes RADIUS Access-Request de la captura de Wireshark, realizad manualmente los cálculos para obtener el valor que debería tener el campo Authenticator en el mensaje RADIUS que se genera como respuesta y comparad éste con el campo correspondiente en la captura.

Comprobad y verificad el correcto funcionamiento de alguno de los EAP Message-Authenticator. Para ello, realizad manualmente los cálculos para obtener este valor de alguno de los paquetes capturados y comparad éste con el campo correspondiente en la captura.

Para el cálculo manual recordad:

```
openssl - OpenSSL command line tool.

$ openssl dgst -md5 -mac hmac -macopt hexkey:<Secret-RADIUS_Hex> file.bin
$ openssl dgst -md5 -mac hmac -macopt key:<Secret-RADIUS_ASCII> file.bin
```

generarían el HMAC-MD5 del fichero *file.bin* utilizando la clave dada en formato hexadecimal o ASCII respectivamente.

3.3 Autenticación basada en EAP-TTLS

Una vez vista la autenticación basada en EAP-MD5 pasaremos a utilizar un método EAP que ofrezca mayores garantías de seguridad y funcionalidades más avanzadas.

Comenzaremos utilizando el método EAP-TTLS. En este método se establece una sesión TLS con el servidor RADIUS que permitirá proteger la autenticación del cliente.

Ejecutad el comando que os permita generar un proceso de autenticación basado en EAP-TTLS.

Capturad el intercambio de paquetes usando Wireshark y analizad el intercambio. ¿Qué ha ocurrido?

Para permitir que el servidor RADIUS maneje autenticaciones basadas en EAP-TTLS es necesario modificar la configuración del servidor. Para ello, **pararemos el servidor y añadiremos en el archivo eap.conf (dentro de los Supported EAP types) lo siguiente**:

```
## EAP-TLS
tls {
      #
      #  These is used to simplify later configurations.
      #
      certdir = /home/alumnos/Asignaturas/SRC/eap/certs
      cadir = / home/alumnos/Asignaturas/SRC/eap/certs

      private_key_file = ${certdir}/radius.key

      #  server certificate.
      certificate_file = ${certdir}/radius.pem

      #  Trusted Root CA list
      CA_file = ${cadir}/ca.pem

      #
      #  For DH cipher suites to work, you have to
      #  run OpenSSL to create the DH file first:
      #
      #       openssl dhparam -out certs/dh 1024
      #
      dh_file = ${certdir}/dh
      random_file = /dev/urandom

      CA_path = ${cadir}

      cipher_list = "DEFAULT"

      verify {
      }
}

## EAP-TTLS
ttls {
      default_eap_type = md5

      copy_request_to_tunnel = no

      use_tunneled_reply = no

      virtual_server = "inner-tunnel"
}
```

Antes de lanzar denuevo el servidor RADIUS es necesario generar los certificados y ficheros que se usan durante el establecimiento de la sesión TLS.

Tal y como se puede observar en el archivo de configuración, es necesario **crear una clave privada RSA de 2048 bits y un certificado de CA y almacenarlos en los ficheros `ca.key` y `ca.pem`, respectivamente, en el directorio `certs`**.

Asimismo, es necesario **generar una clave privada y un certificado para el servidor**. Cread una clave privada RSA de 2048 bits y firmad el CSR asociado con la CA que habéis generado antes. Almacenad en el directorio `certs` tanto la clave privada (a la que daréis el nombre `radius.key`) como el certificado firmado (al que daréis el nombre `radius.pem`).

Para la generación tanto de la CA como de la pareja de claves del servidor, NO es necesario modificar el archivo `openssl.cnf` dado que las opciones por defecto son adecuadas y se puede especificar en la línea de comandos los ficheros de input y output utilizados.

Por último, también es necesario generar un fichero con parámetros Diffie-Hellman que se empleará en caso de que las Cipher Suite empleadas en el establecimiento de sesión TLS utilicen DHE.

Para ello ejecutar el comando:

```
$ openssl dhparam -out dh 2048
```

Colocad el archivo `dh` resultante en el directorio `certs`.

Una vez creados y colocados en el directorio correspondiente todos los ficheros, lanzad denuevo el servidor RADIUS. Si la generación de los certificados y demás ficheros ha sido correcta, el servidor arrancará sin problemas generando un log similar al anterior.

```
Listening on authentication address * port 1812
Listening on authentication address 127.0.0.1 port 18120 as server inner-tunnel
Listening on proxy address * port 1814
Ready to process requests.
```

Una vez que el servidor esté corriendo, será el momento de **ejecutar el comando que os permita generar un proceso de autenticación basado en EAP-TTLS**.

Capturad y analizad el intercambio de paquetes usando Wireshark. Identificad las diferentes fases de la autenticación. Aparte del uso de otro método de autenticación, ¿hay alguna diferencia destacable con el caso anterior?

¿Se puede realizar en este caso la comprobación del EAP-MD5 Challenge / Response?

3.4 Autenticación basada en EAP-TLS

A diferencia del caso EAP-TTLS donde la autenticación del cliente se hace a través de un método EAP que está protegido por la sesión TLS que se establece con el servidor RADIUS, en el método EAP-TLS la autenticación del cliente se realiza aprovechando la funcionalidad de autenticación mutua que permite el establecimiento de sesión TLS.

Para ello, tendremos que generar un certificado para el cliente. G**enerad una clave privada y un certificado para el cliente**. Cread una clave privada RSA de 2048 bits y firmad el CSR asociado con la CA que habéis generado antes. Almacenad en el directorio `certs` tanto la clave privada (a la que daréis el nombre `client.key`) como el certificado firmado (al que daréis el nombre `client.pem`).

Ejecutad el comando que os permita generar un proceso de autenticación basado en EAP-TLS.

En este caso tendréis que utilizar también las opciones:
-k *<user_key_file>*
user certificate key file
-j *<user_cert_file>*
user certificate file

Capturad y analizad el intercambio de paquetes usando Wireshark. Identificad las diferentes fases de la autenticación.

Pedid que algún compañero os firme el CSR de vuestro cliente con su CA y probad a utilizar ese certificado (`client-foreignCA.pem`) en la ejecución de `rad_eap_test` ¿Qué ocurre?.

Se os ocurre alguna otra circunstancia que provoque una situación como la anterior. ¿Cuál? Tratad de generarla.

3.5 Acceso a una red IEEE 802.11 basado en IEEE 802.1X

En el laboratorio se ha desplegado una red IEEE 802.11 como la de la Figura 1. La configuración de seguridad del punto de acceso se ha configurado usando la opción WPA2-Enterprise que hace uso de IEEE 802.1X tanto para permitir el acceso a la red como para generar dinámicamente las claves WPA2. Se trata, hoy por hoy, de la forma más robusta de seguridad en redes IEEE 802.11.

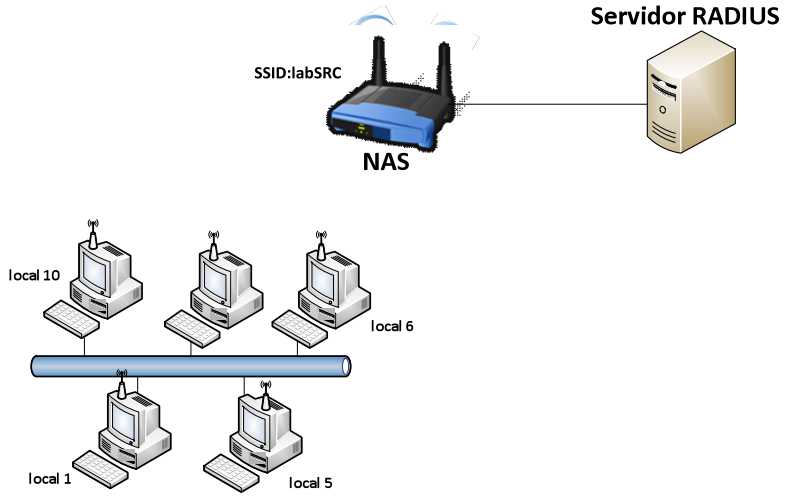

Figura 1: Diagrama de red del laboratorio

Antes de conectarse a la red *labSRC* configuraremos el equipo para poder capturar las tramas a nivel IEEE 802.11. Crearemos un interfaz inalámbrico virtual en modo monitor y comenzaremos a capturar con Wireshark en él.

Para poner el interfaz en modo monitor es necesario ejecutar el siguiente comando:

```
$ sudo iw phy phy0 interface add moni0 type monitor

$ sudo ifconfing moni0 up
```

Para evitar que la vista de la captura de Wireshark se llene con tramas tanto de gestión IEEE 802.11 como EAPOL del resto, se sugiere que se apliquen filtros:

eapol && wlan.addr==11:22:33:44:55:66

Con un filto como el anterior, sólo aparecerían en pantalla las tramas EAPOL que tuvieran esa como dirección MAC de origen o destino.

Una vez lanzada la captura nos conectaremos a la red *labSRC*.

Cuando se selecciona la red *labSRC* aparecen las distintas opciones posibles para la autenticación de WPA & WPA2 Enterprise.

Dada la captura *`radius-labSRC.cap`*, configurad vuestro equipo de forma que vuestra conexión a la red genere una captura análoga. Toda vez que no podéis capturar en el segmento de red entre el NAS y el servidor RADIUS, estableced la correlación con las tramas EAPOL que sí podréis realizar desde vuestro PC.

Identificad y analizad las diferentes fases de la autenticación.

Dependiendo del método de autenticación es necesario introducir diferentes parámetros en el wizard de configuración:

En el caso de EAP-TLS:

Los únicos parámetros obligatorios son el nombre de usuario (`Identity`), la clave privada del usuario (`Private key`) y la clave para acceder a dicha clave (`Private key password`). Incluir el certificado de la CA (`CA certificate`) es opcional.

Es importante destacar que **la clave privada del usuario, deberá pasarse en formato PKCS#12.**

En el caso de EAP-TTLS:

Los únicos parámetros obligatorios son el nombre de usuario (`Anonymous identity` y `Username`) y el password del usuario (`Password`). Incluir el certificado de la CA (`CA certificate`) es opcional.

En el caso de PEAP:

Los únicos parámetros obligatorios son el nombre de usuario (`Anonymous identity` y `Username`) y el password del usuario (`Password`). Incluir el certificado de la CA (`CA certificate`) es opcional.