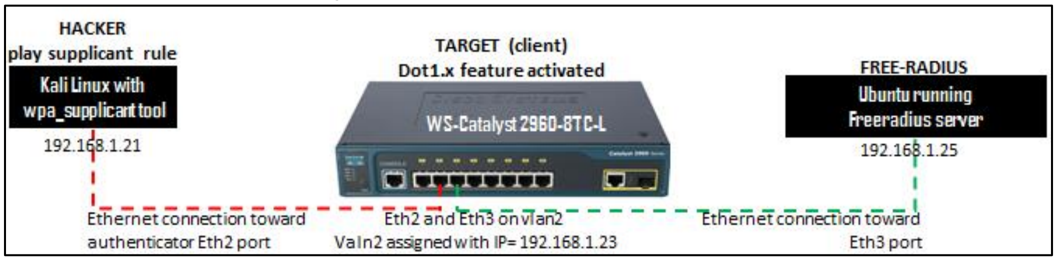
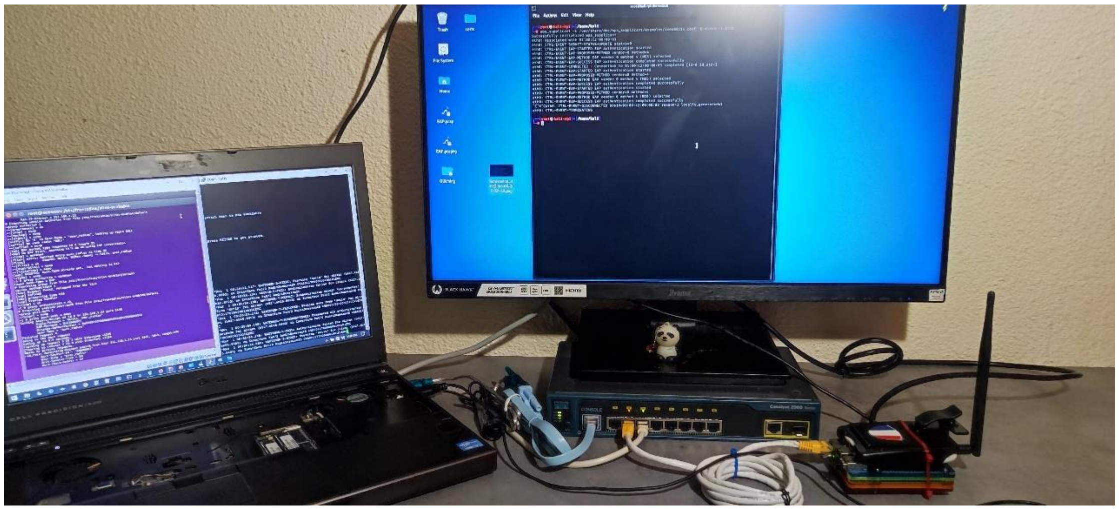
EAPOL-MD5 Fuzzing

Let's take this true-life example:


HACKER
play supplicant rule
Kali Linux with wpa_supplicant tool
192.168.1.21
TARGET (client)
Dot1.x feature activated
WS-Catalyst 2960-8TC-L
FREE-RADIUS
Ubuntu running Freeradius server
192.168.1.25
Ethernet connection toward authenticator Eth2 port
Eth2 and Eth3 on vlan2
Valn2 assigned with IP= 192.168.1.23
Ethernet connection toward Eth3 port

In this example we are going to fuzz the dot1.x authentication feature for a [WS-Catalyst 2960-8TC-L cisco switch]() that support multi-host authentication.



Prepare your lab environment

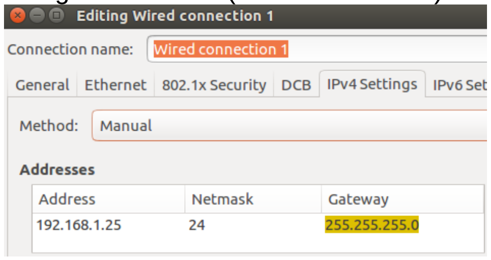
Ubuntu preparation

Install freeradius and then setup the server:

- Edit the "users":

```
# This is an entry for a user with a space in their name.
# Note the double quotes surrounding the name.

"user_radius" Cleartext-Password := "my_pass"
              Reply-Message = "Hello, %{User-Name}"
```

- Assign IP address (ex. 192.168.1.25) to the interface connected to the cisco



Editing Wired connection 1

Connection name: Wired connection 1

General | Ethernet | 802.1x Security | DCB | IPv4 Settings | IPv6 Set

Method: Manual

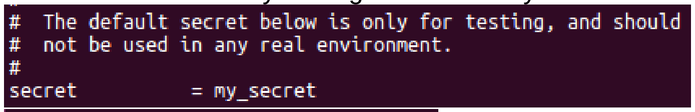
Addresses

Address	Netmask	Gateway
192.168.1.25	24	255.255.255.0

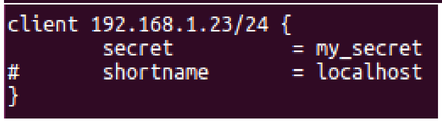
- Edit the "radius.conf" by adding the server IP address:

```
        #  IP address on which to listen.
        #  Allowed values are:
        #       dotted quad (1.2.3.4)
        #       hostname    (radius.example.com)
        #       wildcard    (*)
        ipaddr = 192.168.1.25
```

- Edit the “client.conf” by adding the secret key of the server and the client:

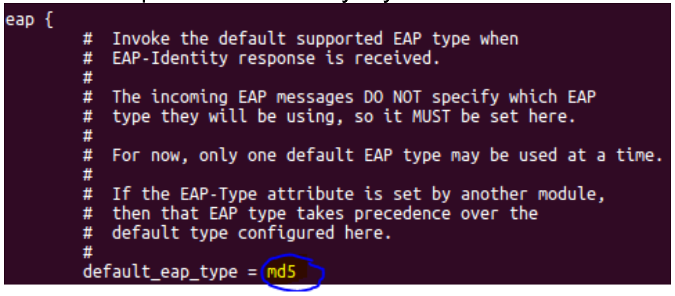


```
#  The default secret below is only for testing, and should
#  not be used in any real environment.
#
secret          = my_secret
```



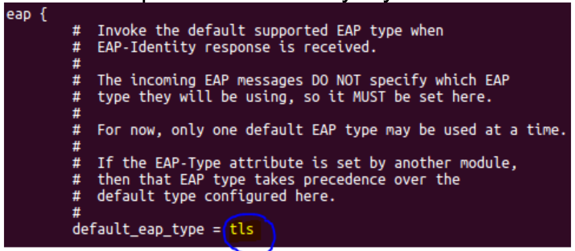
```
client 192.168.1.23/24 {
        secret          = my_secret
#       shortname       = localhost
}
```

- Edit the “eap.conf” in this way if you want the server to only accept MD5 request:

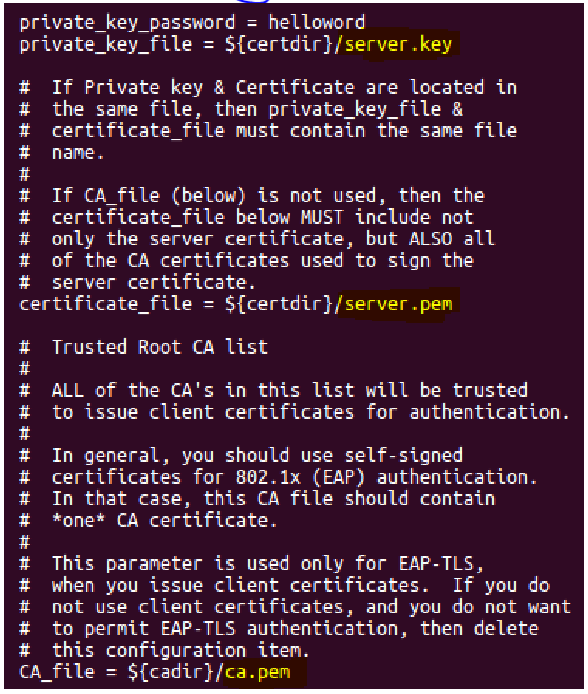


```
eap {
        #  Invoke the default supported EAP type when
        #  EAP-Identity response is received.
        #
        #  The incoming EAP messages DO NOT specify which EAP
        #  type they will be using, so it MUST be set here.
        #
        #  For now, only one default EAP type may be used at a time.
        #
        #  If the EAP-Type attribute is set by another module,
        #  then that EAP type takes precedence over the
        #  default type configured here.
        #
        default_eap_type = md5
```

- Edit the “eap.conf” in this way if you want the server to only accept TLS request:

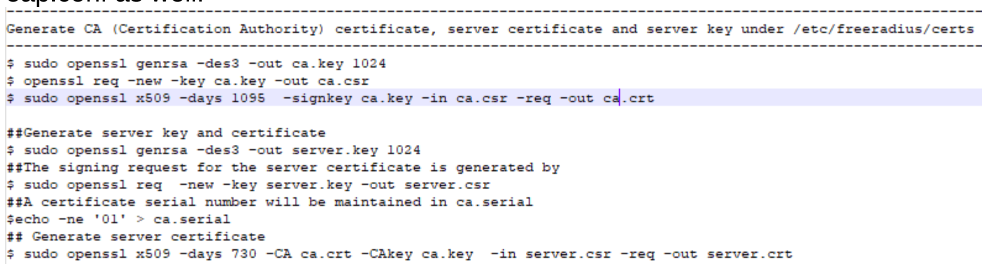


```
eap {
        #  Invoke the default supported EAP type when
        #  EAP-Identity response is received.
        #
        #  The incoming EAP messages DO NOT specify which EAP
        #  type they will be using, so it MUST be set here.
        #
        #  For now, only one default EAP type may be used at a time.
        #
        #  If the EAP-Type attribute is set by another module,
        #  then that EAP type takes precedence over the
        #  default type configured here.
        #
        default_eap_type = tls
```



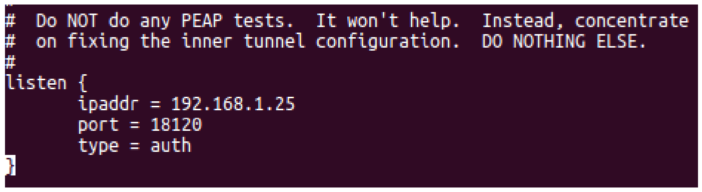
```
private_key_password = helloword
private_key_file = ${certdir}/server.key

#  If Private key & Certificate are located in
#  the same file, then private_key_file &
#  certificate_file must contain the same file
#  name.
#
#  If CA_file (below) is not used, then the
#  certificate_file below MUST include not
#  only the server certificate, but ALSO all
#  of the CA certificates used to sign the
#  server certificate.
certificate_file = ${certdir}/server.pem

#  Trusted Root CA list
#
#  ALL of the CA's in this list will be trusted
#  to issue client certificates for authentication.
#
#  In general, you should use self-signed
#  certificates for 802.1x (EAP) authentication.
#  In that case, this CA file should contain
#  *one* CA certificate.
#
#  This parameter is used only for EAP-TLS,
#  when you issue client certificates.  If you do
#  not use client certificates, and you do not want
#  to permit EAP-TLS authentication, then delete
#  this configuration item.
CA_file = ${cadir}/ca.pem
```

- If you want to generate your own certificates instead of using the existing, do the following and then added them to the “certs” directory and do the change on eap.conf as well:



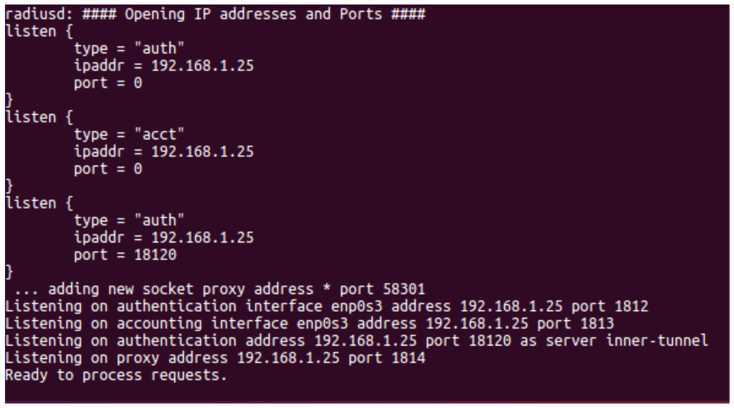
```
-------------------------------------------------------------------------------------------------------------
Generate CA (Certification Authority) certificate, server certificate and server key under /etc/freeradius/certs
-------------------------------------------------------------------------------------------------------------
$ sudo openssl genrsa -des3 -out ca.key 1024
$ openssl req -new -key ca.key -out ca.csr
$ sudo openssl x509 -days 1095  -signkey ca.key -in ca.csr -req -out ca.crt

##Generate server key and certificate
$ sudo openssl genrsa -des3 -out server.key 1024
##The signing request for the server certificate is generated by
$ sudo openssl req  -new -key server.key -out server.csr
##A certificate serial number will be maintained in ca.serial
$echo -ne '01' > ca.serial
## Generate server certificate
$ sudo openssl x509 -days 730 -CA ca.crt -CAkey ca.key  -in server.csr -req -out server.crt
```

- Edit the “inner-tunnel” from the “site-enabled” and “site-available”:



```
#  Do NOT do any PEAP tests.  It won't help.  Instead, concentrate
#  on fixing the inner tunnel configuration.  DO NOTHING ELSE.
#
listen {
       ipaddr = 192.168.1.25
       port = 18120
       type = auth
}
```

- Now, we are ready to run the radius server in debug mode if we want:
  #>freeradius -X



```
radiusd: #### Opening IP addresses and Ports ####
listen {
        type = "auth"
        ipaddr = 192.168.1.25
        port = 0
}
listen {
        type = "acct"
        ipaddr = 192.168.1.25
        port = 0
}
listen {
        type = "auth"
        ipaddr = 192.168.1.25
        port = 18120
}
 ... adding new socket proxy address * port 58301
Listening on authentication interface enp0s3 address 192.168.1.25 port 1812
Listening on accounting interface enp0s3 address 192.168.1.25 port 1813
Listening on authentication address 192.168.1.25 port 18120 as server inner-tunnel
Listening on proxy address 192.168.1.25 port 1814
Ready to process requests.
```

Switch preparation

- Create a vlan (ex. vlan2) and assign to it an IP address (ex. 192.168.1.23)
- Assign both port 2 and 3 on vlan2.
- Enable dot1x multihost on switch [12]:
  Please follow the instruction from this document, we only need the information exist Under this chapter (Configuring 802.1x Readiness Check- page 22) until chapter (Configuring the Host Mode-page28).

Now you are ready to use pyfuzz on your Linux machine as client for your fuzzing.

https://github.com/VraiHack/pyfuzz