Содержание

Example of how to configure FreeRadius3

This section contains the minimal modifications to the FreeRadius3 configuration.

These modifications should be considered only as an example of the FreeRadius configuration. Here we do not address the FreeRadius integration with a billing system or a database.

Suppose, the Radius server has been assigned IP address 192.168.1.200 and port 1812.

VAS Experts dictionary

First you should add a dictionary `dictionary.vasexperts` of vendor-specific attributes to the Radius server dictionary. For this purpose:

- copy the dictionary `/usr/share/dpi/dictionary.vasexperts` from the fastpcrf installation package to the `$freeRadius/share/freeradius/` directory
- add the following line to the main `$freeRadius/share/freeradius/dictionary` `dictionary`:

```
$INCLUDE dictionary.vasexperts
```

FastPCRF instantiating

The parameters to establish connection with the Radius server should be specified in the fastpcrf.configuration file containing the fastPCRF instance (i.e. the Radius client) settings:

```
radius_server=secret123@192.168.1.200%eth0:1812;msg_auth_attr=1
```

Here `eth0` is the local (from the user's point of view) name of the network interface card which is used to establish the connection with the server 192.168.1.200.

Note that the Radius server settings and the client ones should match!

For each fastPCRF instance, the first step is to create a FreeRadius client. Let's name it the `fastdpi1` client. All clients (the fastPCRF instances) will refer to the same virtual fastdpi-vs server.

Add to the Radius server raddb/clients.conf file the following lines:

```
client fastdpi1 {
    ipaddr        = 192.168.1.32
    secret        = secret123
    require_message_authenticator = yes
#   add_cui = yes
    virtual_server    = fastdpi-vs
}
```

Here:

- ipaddr – specifies the fastPCRF instance IP address, it corresponds to 192.168.1.32 in our example
- secret – a unique secret, which is known to the Radius server and client (that is, the fastPCRF instance). The secret string value is user-defined. Note that the same secret is specified in the fastpcrf.conf settings: radius_server=secret123@192.168.1.200%eth0:1812
- require_message_authenticator – a flag specifying that the Radius request should containg the`Message-Authenticator` attribute. RFC 2869 strongly recommends to use this attribute. This setting must be consistent with the `msg_auth_attr` option in the fastpcrf.conf: radius_server=…;msg_auth_attr=1
- add_cui – do not set this option to `yes` value! The Radius server pass the CUI (Chargeable-User-Identity) attribute as the encrypted hash value of the user login for the sake of security. This approach is unacceptable for the fastDPI, so we have to use the user login in plain text. Therefore, `add_cui` is commented out here.
- virtual_server – specifies the name of the virtual server. Its configuration will be considered further.

Creating of the virtual server

In order to configure a virtual server please copy the `raddb/sites-available/default` file which is the part of the FreeRadius installation package to the `raddb/sites-enabled/fastdpi-vs` and then modify the `fastdpi-vs` the following manner:

- set the virtual server name - replace the line `server default` at the beginning of the file with the `server fastdpi-vs` one
- in the listen section for the auth requests (type = auth) specify IP address and port to be listened to (note that it corresponds to the local Radius server address):

```
ipaddr = 192.168.1.200
port = 1812
interface = eth0
```

- the rest sections of the `listen` one are to be deleted (or commented out – we don't need them)
- all the main steps to form a response to Access-Request are specified within the post-auth section. It is impossible to provide any guidance because it relates the provider and the Radius server environment dependent information. A list of required attributes is described within the "RADIUS ACCESS-ACCEPT" section. As an example, the statically defined attributes of the Access-Accept response are given (note that if the CUI (Chargeable-User-Identity) attribute containing a single zero byte is used in the Access-Request, it indicates that the fastPCRF does not know the user login and have to request it from the Radius server. The CUI is formed from the Framed-IP-Address only as an illustration for this example):

```
post-auth {
…
    #
    # Add VasExperts attributes
    #
    if ( Chargeable-User-Identity == 0x00 ) {
      update reply {
        Chargeable-User-Identity := "u-%{Framed-IP-Address}"
      }
    }
    else {
       update reply {
         Chargeable-User-Identity := "%{Chargeable-User-Identity}"
        }
    }
    update reply {
        Framed-IP-Address := "%{Framed-IP-Address}"
        VasExperts-Policing-Profile := "test1"
        VasExperts-Service-Profile  += "1:test1"
        Session-Timeout := 300
    }
…
}
```

- The cui option of the post-auth section is to be left commented out! FreeRadius passes a login hash value instead of the plain user login to the CUI. We should avoid that, so we'll create the CUI attribute withing the response according to the example above.
- Then add to the Post-Auth-Type REJECT section (Access-Reject forming) add following items
    - CUI attribute in case the fastPCRF request it and the user is known;
    - VasExperts-Policing-Profile attribute specifying the policing profile used by unauthorized users (in the example below the profile name is `plc_unauth`, you should use another name according to your settings);
    - VasExperts-Service-Profile attribute specifying the service 5 profile (“Whitelist”). Typically this profile allows unauthorized users access just the Captive Portal. In the example below the profile name is `cp_unauth`, you should use another name according to your settings.

Example:

```
if (Chargeable-User-Identity == "\0" ) {
   update reply {
    Chargeable-User-Identity := "login"
   }
}
update reply {
    VasExperts-Policing-Profile := "plc_unauth"
    VasExperts-Service-Profile  += "5:cp_unauth"
}
```

Users editing

You should add two fastPCRF entries to the raddb/users file:

```
VasExperts.FastDPI.unknownUser Cleartext-Password := "VasExperts.FastDPI"
DEFAULT Cleartext-Password := "VasExperts.FastDPI"
```

The first entry specifies user name being sent by the fastPCRF in case of unknown login, more details see the `radius_unknown_user` configuration option description. The name is specified in the fastPCRF, as well as password, more details see the `radius_unknown_user_psw` configuration option description. The second entry specifies the password used by the fastPCRF to send requests for known logins. This password is specified in the fastPCRF, more details see the `radius_user_password` configuration option description.