Auth_radius Module

Jan Janak
FhG Fokus

Juha Heinanen
Song Networks

Stelios Sidiroglou-Douskos

Edited by

Jan Janak

Auth_radius Module

Edited by Jan Janak, Juha Heinanen, and Stelios Sidiroglou-Douskos,and Jan Janak

Revision History

Revision $Revision: 1.3 $ $Date: 2003/07/14 21:19:21 $

Table of Contents

List of Examples

Chapter 1. User's Guide

1.1. Overview

This module contains functions that are used to perform authentication using a Radius server. Basically the proxy will pass along the credentials to the radius server which will in turn send a reply containing result of the authentication. So basically the whole authentication is done in the Radius server. Before sending the request to the radius server we perform some sanity checks over the credentials to make sure that only well formed credentials will get to the server. We have implemented radius authentication according to draft-sterman-aaa-sip-00. This module requires radiusclient library version 0.4.1 or higher which is available from http://developer.berlios.de/projects/radiusclient-ng/ (http://developer.berlios.de/projects/radiusclient-ng/).

How to configure radius server -- more detailed description -- TBD.

> **Warning**
>
> The detailed description of radius authentication setup is important since many people will use it and we want to make the setup painless.

1.2. Dependencies

The module depends on the following modules (in the other words the listed modules must be loaded before this module):

- *auth* -- Generic authentication functions

1.3. Exported Parameters

1.3.1. `radius_config` (string)

This is the location of the configuration file of radius client libraries.

Default value is "/usr/local/etc/radiusclient/radiusclient.conf".

Example 1-1. `radius_config` parameter usage

```
modparam("auth_radius", "radius_config", "/etc/radiusclient.conf")
```

1.3.2. `service_type` (integer)

This is the value of the Service-Type radius attribute to be used. The default should be fine for most people. See your radius client include files for numbers to be put in this parameter if you need to change it.

Default value is "15".

Example 1-2. `radius_config` usage

```
modparam("auth_radius", "service_type", 15)
```

1.4. Exported Functions

1.4.1. `radius_www_authorize(realm)`

The function verifies credencials according to RFC2617. If the credentials are verified sucessfully then the function will succeed and mark the credentials as authorized (marked credentials can be later used by some other functions). If the function was unable to verify the credentials for some reason then it will fail and the script should call `www_challenge` which will challenge the user again.

This function will, in fact, perform sanity checks over the received credentials and then pass them along to the radius server which will verify the credentials and return whether they are valid or not.

Meaning of the parameter is as follows:

- *realm* - Realm is a opaque string that the user agent should present to the user so he can decide what username and password to use. Usualy this is domain of the host the server is running on.

  If an empty string "" is used then the server will generate it from the request. In case of REGISTER requests To header field domain will be used (because this header field represents a user being registered), for all other messages From header field domain will be used.

Example 1-3. `radius_www_authorize` usage

```
...
if (radius_www_authorize("iptel.org")) {
    www_challenge("iptel.org", "1");
};
...
```

1.4.2. `radius_proxy_authorize(realm)`

The function verifies credencials according to RFC2617. If the credentials are verified sucessfully then the function will succeed and mark the credentials as authorized (marked credentials can be later used by some other functions). If the function was unable to verify the credentials for some reason then it will fail and the script should call `proxy_challenge` which will challenge the user again.

This function will, in fact, perform sanity checks over the received credentials and then pass them along to the radius server which will verify the credentials and return whether they are valid or not.

Meaning of the parameter is as follows:

- *realm* - Realm is a opaque string that the user agent should present to the user so he can decide what username and password to use. Usualy this is domain of the host the server is running on.

  If an empty string "" is used then the server will generate it from the request. From header field domain will be used as realm.

Example 1-4. proxy_authorize usage

```
...
if (!radius_proxy_authorize("")) {
    proxy_challenge("", "1");  # Realm will be autogenerated
};
...
```

Chapter 2. Developer’s Guide

To be done.

Chapter 3. Frequently Asked Questions

1. What is the meaning of life ?

42