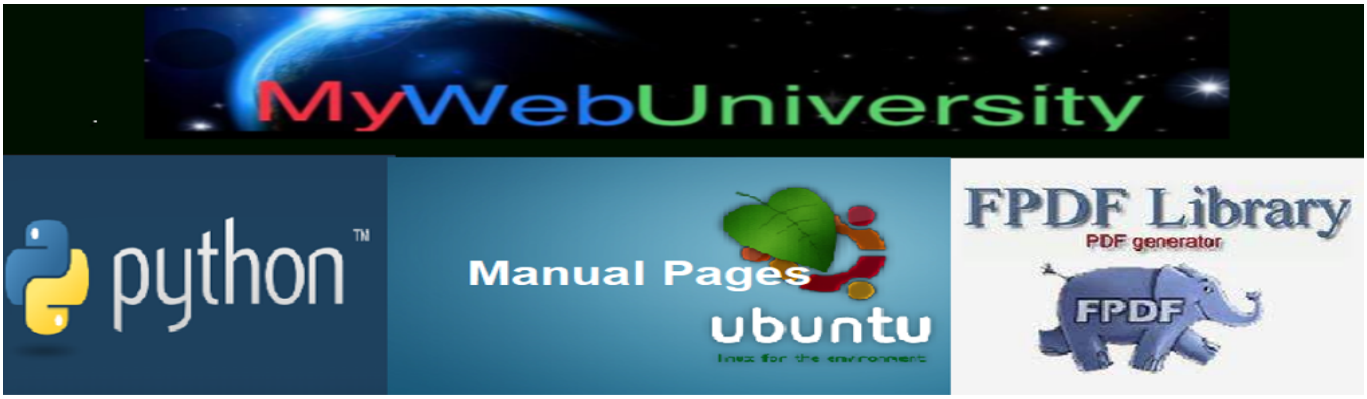

Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'Authen::Radius.3pm'

$ man Authen::Radius.3pm

Radius(3pm) User Contributed Perl Documentation Radius(3pm)

NAME

Authen::Radius - provide simple Radius client facilities

SYNOPSIS

```
use Authen::Radius;

$r = new Authen::Radius(Host => 'myserver', Secret => 'mysecret');
print "auth result=", $r->check_pwd('myname', 'mypwd'), "\n";

$r = new Authen::Radius(Host => 'myserver', Secret => 'mysecret');
Authen::Radius->load_dictionary();
$r->add_attributes (
    { Name => 'User-Name', Value => 'myname' },
    { Name => 'Password', Value => 'mypwd' },
# RFC 2865 http://www.ietf.org/rfc/rfc2865.txt calls this attribute
# User-Password. Check your local RADIUS dictionary to find
# out which name is used on your system
#    { Name => 'User-Password', Value => 'mypwd' },
    { Name => 'h323-return-code', Value => '0' }, # Cisco AV pair
    { Name => 'Digest-Attributes', Value => { Method => 'REGISTER' } }
);
```

```
$r->send_packet(ACCESS_REQUEST) and $type = $r->recv_packet();
print "server response type = $type\n";
for $a ($r->get_attributes()) {
  print "attr: name=$a->{'Name'} value=$a->{'Value'}\n";
}
```

DESCRIPTION

The "Authen::Radius" module provides a simple class that allows you to send/receive Radius requests/responses to/from a Radius server.

CONSTRUCTOR

new (Host => HOST, Secret => SECRET [, TimeOut => TIMEOUT] [,Service => SERVICE] [, Debug => Bool] [, LocalAddr => hostname[:port]] [,Rfc3579MessageAuth => Bool] [,NodeList= NodeListArrayRef])

Creates & returns a blessed reference to a Radius object, or undef on failure. Error status may be retrieved with "Authen::Radius::get_error" (errorcode) or "Authen::Radius::strerror" (verbose error string).

The default "Service" is "radius", the alternative is "radius-acct". If you do not specify port in the "Host" as a "hostname:port", then port specified in your /etc/services will be used. If there is nothing there, and you did not specify port either then default is 1812 for "radius" and 1813 for "radius-acct".

Optional parameter "Debug" with a Perl "true" value turns on debugging (verbose mode).

Optional parameter "LocalAddr" may contain local IP/host bind address from which RADIUS packets are sent.

Optional parameter "Rfc3579MessageAuth" with a Perl "true" value turns on generating of Message-Authenticator for Access-Request (RFC3579, section 3.2). The Message-Authenticator is always generated for Status-Server packets.

Optional parameter "NodeList" may contain a Perl reference to an array, containing a

list of Radius Cluster nodes. Each nodes in the list can be specified using a hostname or IP (with an optional port number), i.e. 'radius1.mytel.com' or 'radius.myhost.com:1812'. Radius Cluster contains a set of Radius servers, at any given moment of time only one server is considered to be "active" (so requests are send to this server). How the active node is determined? Initially in addition to the "NodeList" parameter you may supply the "Host" parameter and specify which server should become the first active node. If this parameter is absent, or the current active node does not reply anymore, the process of "discovery" will be performed: a request will be sent to all nodes and the consecutive communication continues with the node, which will be the first to reply.

METHODS

load_dictionary ([DICTIONARY], [format => 'freeradius' | 'gnuradius'])

Loads the definitions in the specified Radius dictionary file (standard Livingston radiusd format). Tries to load "/etc/raddb/dictionary" when no argument is specified, or dies. "format" should be specified if dictionary has other format (currently supported: FreeRADIUS and GNU Radius)

NOTE: you need to load valid dictionary if you plan to send RADIUS requests with attributes other than just "User-Name"/"Password".

check_pwd (USERNAME, PASSWORD [,NASIPADDRESS])

Checks with the RADIUS server if the specified "PASSWORD" is valid for user "USERNAME". Unless "NASIPADDRESS" is specified, the script will attempt to determine it's local IP address (IP address for the RADIUS socket) and this value will be placed in the NAS-IP-Address attribute. This method is actually a wrapper for subsequent calls to "clear_attributes", "add_attributes", "send_packet" and "recv_packet". It returns 1 if the "PASSWORD" is correct, or undef otherwise.

add_attributes ({ Name => NAME, Value => VALUE [, Type => TYPE] [, Vendor => VENDOR] [, Tag => TAG] }, ...)

Adds any number of Radius attributes to the current Radius object. Attributes are specified as a list of anon hashes. They may be "Name"d with their dictionary name

(provided a dictionary has been loaded first), or with their raw Radius attribute-type values. The "Type" pair should be specified when adding attributes that are not in the dictionary (or when no dictionary was loaded). Values for "TYPE" can be "'string'", "'integer'", "'ipaddr'", "'ipv6addr'", "'ipv6prefix'", "'ifid'" or "'avpair'". The "VENDOR" may be Vendor's name from the dictionary or their integer id. For tagged attributes (RFC2868) tag can be specified in "Name" using 'Name:Tag' format, or by using "Tag" pair. TAG value is expected to be an integer, within [1:31] range (zero value isn't supported).

get_attributes

Returns a list of references to anon hashes with the following key/value pairs : { Name => NAME, Code => RAWTYPE, Value => VALUE, RawValue => RAWVALUE, Vendor => VENDOR, Tag => TAG, AttrName => NAME }. Each hash represents an attribute in the current object. The "Name" and "Value" pairs will contain values as translated by the dictionary (if one was loaded). The "Code" and "RawValue" pairs always contain the raw attribute type & value as received from the server. If some attribute doesn't exist in dictionary or type of attribute not specified then corresponding "Value" undefined and "Name" set to attribute ID ("Code" value). For tagged attribute (RFC2868), it will include the tag into the "NAME" as 'Name:Tag'. Original Name is stored in "AttrName". Also value of tag is stored in "Tag" (undef for non-tagged attributes).

clear_attributes

Clears all attributes for the current object.

send_packet (REQUEST_TYPE, RETRANSMIT)

Packs up a Radius packet based on the current secret & attributes and sends it to the server with a Request type of "REQUEST_TYPE". Exported "REQUEST_TYPE" methods are "ACCESS_REQUEST", "ACCESS_ACCEPT", "ACCESS_REJECT", "ACCESS_CHALLENGE", "ACCOUNTING_REQUEST", "ACCOUNTING_RESPONSE", "ACCOUNTING_STATUS", "STATUS_SERVER", "DISCONNECT_REQUEST", "DISCONNECT_ACCEPT", "DISCONNECT_REJECT", "COA_REQUEST", "COA_ACCEPT", "COA_REJECT", "COA_ACK", and "COA_NAK". Returns the number of bytes sent, or undef on failure.

If the RETRANSMIT parameter is provided and contains a non-zero value, then it is considered that we are re-sending the request, which was already sent previously. In this case the previous value of packet identifier is used.

recv_packet (DETECT_BAD_ID)

Receives a Radius reply packet. Returns the Radius Reply type (see possible values for "REQUEST_TYPE" in method "send_packet") or undef on failure. Note that failure may be due to a failed recv() or a bad Radius response authenticator. Use "get_error" to find out.

If the DETECT_BAD_ID parameter is supplied and contains a non-zero value, then calculation of the packet identifier is performed before authenticator check and EBADID error returned in case when packet identifier from the response doesn't match to the request. If the DETECT_BAD_ID is not provided or contains zero value then EBADAUTH returned in such case.

set_timeout (TIMEOUT)

Sets socket I/O activity timeout. "TIMEOUT" should be specified in floating seconds since the epoch.

get_error

Returns the last "ERRORCODE" for the current object. Errorcodes are one-word strings always beginning with an "'E"'.

strerror ([ERRORCODE])

Returns a verbose error string for the last error for the current object, or for the specified "ERRORCODE".

error_comment

Returns the last error explanation for the current object. Error explanation is generated by system call.

get_active_node

Returns currently active radius node in standard numbers-and-dots notation with port delimited by colon.

AUTHOR

Carl Declerck <carl@miskatonic.inbe.net> - original design Alexander Kapitanenko <kapitan at portaone.com> and Andrew Zhilenko <andrew at portaone.com> - later modifications.

PortaOne Development Team <perl-radius at portaone.com> is the current module's maintainer at CPAN.