FreeRADIUS

```
--- freeradius-original-no-comments/3.0/radiusd.conf
# correct_escapes = true may or may not be needed

-	auth = no
+	auth = yes
 	auth_badpass = no
 	auth_goodpass = no
+	msg_goodpass = "msg_goodpass: nas: %{NAS-IP-Address}"
+	msg_badpass = "msg_badpass: nas: %{NAS-IP-Address}"



--- freeradius-original-no-comments/3.0/users
 DEFAULT	Hint == "SLIP"
 	Framed-Protocol = SLIP
+DEFAULT
+	Message-Authenticator = 0



--- freeradius-original-no-comments/3.0/mods-config/files/authorize
 	Framed-Compression = Van-Jacobson-TCP-IP
 DEFAULT	Hint == "SLIP"
 	Framed-Protocol = SLIP
+DEFAULT
+	Message-Authenticator = 0



--- freeradius-original-no-comments/3.0/sites-enabled/inner-tunnel
# change all -sql to sql
```

Message-Authenticator and Mikrotik

As of RouterOS version 7.15 changelog, Mikrotik introduced the following two changes:

```
*) radius - added "require-message-auth" option that requires "Message-Authenticator" in receive
*) radius - include "Message-Authenticator" in any RADIUS communication messages besides account
```

When you upgrade from a previous version, currently Mikrotik sets the require-message-auth=yes instead of no. This means that if you're running FreeRADIUS, most likely you won't be able to login to your routers anymore using RADIUS authentciation. I hope you know what the local credentials are!

It's taken much too long to learn how to get FreeRADIUS to add the Message-Authenticator attribute in response messages.

```
# Ubuntu 22.04 - /etc/freeradius/3.0/mods-config/files/authorize
# Add the following to the end of the file, and make sure you're not breaking anything else in
the process
DEFAULT
        Message-Authenticator = 0
```

Using environment variables with Docker

This is an example of using the string String Expansion Alternation syntax with environment variables. If the environment variable doesn't exist, it will return the value sqlite instead.

```
sql {
	dialect = "%{$ENV{RADIUSD_SQL_DIALECT}:-sqlite}"
	driver = "rlm_sql_${dialect}"
```

Here's another example:

```
# a bunch of random examples so you get the idea
type = %{$ENV{RADIUSD_SERVER_DEFAULT_TYPE}:-auth}
private_key_password = %{$ENV{RADIUSD_MOD_EAP_TLS_PRIVATE_KEY_PASSWORD}:-
YouReallyNeedThisEnvVariable}
private_key_file = %{$ENV{RADIUSD_MOD_EAP_TLS_PRIVATE_KEY_FILE}:-
/opt/freeradius/certs/server.key}
certificate_file = %{$ENV{RADIUSD_MOD_EAP_TLS_CERTIFICATE_FILE}:-
```

```
/opt/freeradius/certs/server.crt}
tls_min_version = "%{$ENV{RADIUSD_MOD_EAP_TLS_MIN_VERSION}:-1.2}"
tls_max_version = "%{$ENV{RADIUSD_MOD_EAP_TLS_MAX_VERSION}:-1.2}"
enable = %{$ENV{RADIUSD_MOD_EAP_TLS_OCSP_ENABLE}:-no}
override_cert_url = %{$ENV{RADIUSD_MOD_EAP_TLS_OCSP_OVERRIDE_CERT_URL}:-no}
```

Configs

Environment setup

```
mkdir -p /opt/freeradius/certs
mkdir -p /opt/freeradius/certs/trusted
mkdir -p /opt/freeradius/db
mkdir -p /opt/freeradius/tmp
chown -R freerad:freerad /opt/freeradius
find /opt/freeradius -type d -exec chmod 770 {} \;
```

All files that need updating

```
radiusd.conf
clients.conf
users
mods-config/files/authorize
mods-enabled/linelog
mods-enabled/python3
mods-enabled/sql
mods-enabled/eap
sites-enabled/default
sites-enabled/inner-tunnel
```

Base changes

```
--- freeradius-original-no-comments/3.0/radiusd.conf
# correct_escapes = true may or may not be needed

-	auth = no
+	auth = yes
 	auth_badpass = no
 	auth_goodpass = no
+	msg_goodpass = "msg_goodpass: nas: %{NAS-IP-Address}"
+	msg_badpass = "msg_badpass: nas: %{NAS-IP-Address}"
```

```
--- freeradius-original-no-comments/3.0/users
 DEFAULT	Hint == "SLIP"
 	Framed-Protocol = SLIP
+DEFAULT
+	Message-Authenticator = 0
```

```
--- freeradius-original-no-comments/3.0/mods-config/files/authorize
 	Framed-Compression = Van-Jacobson-TCP-IP
 DEFAULT	Hint == "SLIP"
 	Framed-Protocol = SLIP
+DEFAULT
+	Message-Authenticator = 0
```

```
--- freeradius-original-no-comments/3.0/sites-enabled/inner-tunnel
# change all -sql to sql
```

```
--- freeradius-original-no-comments/3.0/sites-enabled/default
# change all -sql to sql, and a number of other items as needed
#
@@ -1,6 +1,6 @@
 server default {
 listen {
-	type = auth
+	type = auth+acct
 	ipaddr = *
 	port = 0
 	limit {
@@ -34,8 +34,10 @@
 	}
```

```
 }
 authorize {
+␣linelog_recv_request
 ␣filter_username
 ␣preprocess
+␣auth_log
 ␣chap
 ␣mschap
 ␣digest
@@ -44,7 +46,7 @@
 ␣␣ok = return
 ␣}
 ␣files
-␣-sql
+␣sql
 ␣-ldap
 ␣expiration
 ␣logintime
@@ -75,12 +77,12 @@
 }
 accounting {
 ␣detail
-␣unix
-␣-sql
+␣sql
 ␣exec
 ␣attr_filter.accounting_response
 }
 session {
+␣sql
 }
 post-auth {
 ␣if (session-state:User-Name && reply:User-Name && request:User-Name && (reply:User-Name ==
request:User-Name)) {
@@ -91,12 +93,16 @@
 ␣update {
 ␣␣&reply: += &session-state:
 ␣}
-␣-sql
+␣sql_session_start
```

```
+□reply_log
+□linelog_send_accept
+□sql
 □exec
 □remove_reply_message_if_eap
 □Post-Auth-Type REJECT {
-□□-sql
+□□sql
 □□attr_filter.access_reject
+                linelog_send_reject
 □□eap
 □□remove_reply_message_if_eap
 □}
@@ -111,8 +117,12 @@
 □}
 }
 pre-proxy {
+□linelog_send_proxy_request
+□pre_proxy_log
 }
 post-proxy {
+□linelog_recv_proxy_response
+□post_proxy_log
 □eap
 }
 }
```

```
# file: mods-available/linelog
#
# Added based on this URL:
# https://wiki.freeradius.org/guide/eduroam

linelog linelog_recv_request {
        filename = syslog
        syslog_facility = local0
        syslog_severity = debug
        format = '{"action":"Recv-Request", "User-Name":"%{User-Name}", "Called-Station-
Id":"%{Called-Station-Id}", "EAP-Type":"%{EAP-Type}", "NAS-IP-Address":"%{NAS-IP-Address}",
"Calling-Station-Id":"%{Calling-Station-Id}", "Message-Authenticator":"%{Message-
Authenticator}", "NAS-Port-Type":"%{NAS-Port-Type}", "NAS-Port-Id":"%{NAS-Port-Id}", "TLS-
```

```
Client-Cert-Subject-Alt-Name-Dns":"%{TLS-Client-Cert-Subject-Alt-Name-Dns}", "TLS-Client-Cert-
Serial":"%{TLS-Client-Cert-Serial}", "TLS-Client-Cert-Issuer":"%{TLS-Client-Cert-Issuer}"}'
}

linelog linelog_send_accept {
        filename = syslog
        syslog_facility = local0
        syslog_severity = info
        format = '{"action":"Send-Accept", "User-Name":"%{User-Name}", "Called-Station-
Id":"%{Called-Station-Id}", "EAP-Type":"%{EAP-Type}", "NAS-IP-Address":"%{NAS-IP-Address}",
"Calling-Station-Id":"%{Calling-Station-Id}", "Message-Authenticator":"%{Message-
Authenticator}", "NAS-Port-Type":"%{NAS-Port-Type}", "NAS-Port-Id":"%{NAS-Port-Id}", "TLS-
Client-Cert-Subject-Alt-Name-Dns":"%{TLS-Client-Cert-Subject-Alt-Name-Dns}", "TLS-Client-Cert-
Serial":"%{TLS-Client-Cert-Serial}", "TLS-Client-Cert-Issuer":"%{TLS-Client-Cert-Issuer}"}'
}

linelog linelog_send_reject {
        filename = syslog
        syslog_facility = local0
        syslog_severity = error
        format = '{"action":"Send-Reject", "User-Name":"%{User-Name}", "Called-Station-
Id":"%{Called-Station-Id}", "EAP-Type":"%{EAP-Type}", "NAS-IP-Address":"%{NAS-IP-Address}",
"Calling-Station-Id":"%{Calling-Station-Id}", "Message-Authenticator":"%{Message-
Authenticator}", "NAS-Port-Type":"%{NAS-Port-Type}", "NAS-Port-Id":"%{NAS-Port-Id}", "TLS-
Client-Cert-Subject-Alt-Name-Dns":"%{TLS-Client-Cert-Subject-Alt-Name-Dns}", "TLS-Client-Cert-
Serial":"%{TLS-Client-Cert-Serial}", "TLS-Client-Cert-Issuer":"%{TLS-Client-Cert-Issuer}"}'
}

linelog linelog_send_proxy_request {
        filename = syslog
        syslog_facility = local0
        syslog_severity = debug
        format = "action = Send-Proxy-Request, %{pairs:proxy-request:}"
}

linelog linelog_recv_proxy_response {
        filename =
syslog
                            syslog_facility =
local0
```

```
                        syslog_severity = debug
        reference = "messages.%{proxy-reply:Response-Packet-Type}"
        messages {
                Access-Accept = "action = Recv-Proxy-Accept, User-Name = %{User-Name},
Calling-Station-Id = %{Calling-Station-Id}, %{pairs:proxy-reply:}"
                Access-Reject = "action = Recv-Proxy-Reject, User-Name = %{User-Name},
Calling-Station-Id = %{Calling-Station-Id}, %{pairs:proxy-reply:}"
                Access-Challenge = "action = Recv-Proxy-Challenge, User-Name = %{User-Name},
Calling-Station-ID = %{Calling-Station-Id}, %{pairs:proxy-reply:}"
        }
}
```

Solution specific changes

EAP-TLS

```
# file: mods-enabled/eap
eap {
    default_eap_type = tls
    timer_expire = 60
    ignore_unknown_eap_types = no
    cisco_accounting_username_bug = no
    max_sessions = ${max_requests}
    tls-config tls-common {
        private_key_file = /opt/freeradius/certs/freerad1.domain.com.key
        certificate_file = /opt/freeradius/certs/freerad1.domain.com.crt
        ca_file = /opt/freeradius/certs/root-ca-certificate.crt
        check_crl = no
        check_all_crl = no
        ca_path = /opt/freeradius/certs/trusted
        ca_path_reload_interval = 3600
        allow_expired_crl = no
        cipher_list = "DEFAULT"
        cipher_server_preference = yes
        tls_min_version = "1.2"
        tls_max_version = "1.2"
        ecdh_curve = ""
        cache {
```

```
            enable = no
            lifetime = 24 # hours
            store {
                Tunnel-Private-Group-Id
            }
        }
        verify {
        }
        ocsp {
            enable = no
            override_cert_url = no
            use_nonce = yes
        }
    }
    tls {
        tls = tls-common
    }
}
```

SQLite

```
# file: mods-enabled/sql
# note: all other dialect references have been removed
sql {
	dialect = "sqlite"
	driver = "rlm_sql_${dialect}"
	sqlite {
		filename = "/opt/freeradius/db/freeradius.sqlite3"
		busy_timeout = 200
		bootstrap = "${modconfdir}/${..:name}/main/sqlite/schema.sql"
	}
	radius_db = "radius"
	acct_table1 = "radacct"
	acct_table2 = "radacct"
	postauth_table = "radpostauth"
	authcheck_table = "radcheck"
	groupcheck_table = "radgroupcheck"
	authreply_table = "radreply"
```

```
	groupreply_table = "radgroupreply"
	usergroup_table = "radusergroup"
	delete_stale_sessions = yes
	pool {
		start = ${thread[pool].start_servers}
		min = ${thread[pool].min_spare_servers}
		max = ${thread[pool].max_servers}
		spare = ${thread[pool].max_spare_servers}
		uses = 0
		retry_delay = 30
		lifetime = 0
		idle_timeout = 60
	}
	client_table = "nas"
	group_attribute = "SQL-Group"
	$INCLUDE ${modconfdir}/${.:name}/main/${dialect}/queries.conf
}
```

Original Configs

Just for reference... from a system running Ubuntu 24.04.3 freeradius 3.2.5+dfsg-3~ubuntu24.04.3

freeradius/3.0

```
# file: radiusd.conf
prefix = /usr
exec_prefix = /usr
sysconfdir = /etc
localstatedir = /var
sbindir = ${exec_prefix}/sbin
logdir = /var/log/freeradius
raddbdir = /etc/freeradius/3.0
radacctdir = ${logdir}/radacct
```

```
name = freeradius
confdir = ${raddbdir}
modconfdir = ${confdir}/mods-config
certdir = ${confdir}/certs
cadir   = ${confdir}/certs
run_dir = ${localstatedir}/run/${name}
db_dir = ${raddbdir}
libdir = /usr/lib/freeradius
pidfile = ${run_dir}/${name}.pid
max_request_time = 30
cleanup_delay = 5
max_requests = 16384
hostname_lookups = no
log {
	destination = files
	colourise = yes
	file = ${logdir}/radius.log
	syslog_facility = daemon
	stripped_names = no
	auth = no
	auth_badpass = no
	auth_goodpass = no
	msg_denied = "You are already logged in - access denied"
}
checkrad = ${sbindir}/checkrad
ENV {
}
security {
	user = freerad
	group = freerad
	allow_core_dumps = no
	max_attributes = 200
	reject_delay = 1
	status_server = yes
	require_message_authenticator = auto
	limit_proxy_state = auto
}
proxy_requests  = yes
$INCLUDE proxy.conf
$INCLUDE clients.conf
```

```
thread pool {
	start_servers = 5
	max_servers = 32
	min_spare_servers = 3
	max_spare_servers = 10
	max_requests_per_server = 0
	auto_limit_acct = no
}
modules {
	$INCLUDE mods-enabled/
}
instantiate {
}
policy {
	$INCLUDE policy.d/
}
$INCLUDE sites-enabled/
```

```
# file: clients.conf
client localhost {
	ipaddr = 127.0.0.1
	proto = *
	secret = testing123
	nas_type	= other	# localhost isn't usually a NAS...
	limit {
		max_connections = 16
		lifetime = 0
		idle_timeout = 30
	}
}
client localhost_ipv6 {
	ipv6addr	= ::1
	secret		= testing123
}
```

```
# file: proxy.conf
proxy server {
	default_fallback = no
}
home_server localhost {
	type = auth
	ipaddr = 127.0.0.1
	port = 1812
	secret = testing123
	response_window = 20
	zombie_period = 40
	revive_interval = 120
	status_check = status-server
	check_interval = 30
	check_timeout = 4
	num_answers_to_alive = 3
	max_outstanding = 65536
	coa {
		irt = 2
		mrt = 16
		mrc = 5
		mrd = 30
	}
	limit {
	     max_connections = 16
	     max_requests = 0
	     lifetime = 0
	     idle_timeout = 0
	}
}
home_server_pool my_auth_failover {
	type = fail-over
	home_server = localhost
}
realm example.com {
	auth_pool = my_auth_failover
}
realm LOCAL {
}
```

freeradius/3.0/mods-available

```
# file: mods-available/eap
eap {
	default_eap_type = md5
	timer_expire = 60
	ignore_unknown_eap_types = no
	cisco_accounting_username_bug = no
	max_sessions = ${max_requests}
	md5 {
	}
	gtc {
		auth_type = PAP
	}
	tls-config tls-common {
		private_key_password = whatever
		private_key_file = /etc/ssl/private/ssl-cert-snakeoil.key
		certificate_file = /etc/ssl/certs/ssl-cert-snakeoil.pem
		ca_file = /etc/ssl/certs/ca-certificates.crt
		ca_path = ${cadir}
		cipher_list = "DEFAULT"
		cipher_server_preference = no
		tls_min_version = "1.2"
		tls_max_version = "1.2"
		ecdh_curve = ""
		cache {
			enable = no
			lifetime = 24 # hours
			store {
				Tunnel-Private-Group-Id
			}
		}
		verify {
		}
		ocsp {
			enable = no
			override_cert_url = yes
```

```
□□□url = "http://127.0.0.1/ocsp/"
□□}
□}
□tls {
□□tls = tls-common
□}
□ttls {
□□tls = tls-common
□□default_eap_type = md5
□□copy_request_to_tunnel = no
□□use_tunneled_reply = no
□□virtual_server = "inner-tunnel"
□}
□peap {
□□tls = tls-common
□□default_eap_type = mschapv2
□□copy_request_to_tunnel = no
□□use_tunneled_reply = no
□□virtual_server = "inner-tunnel"
□}
□mschapv2 {
□}
}
```

freeradius/3.0/sites-available

```
# file: sites-available/default
server default {
listen {
□type = auth
□ipaddr = *
□port = 0
□limit {
□     max_connections = 16
□     lifetime = 0
□     idle_timeout = 30
□}
}
```

```
listen {
	ipaddr = *
	port = 0
	type = acct
	limit {
	}
}
listen {
	type = auth
	ipv6addr = ::	# any.  ::1 == localhost
	port = 0
	limit {
	      max_connections = 16
	      lifetime = 0
	      idle_timeout = 30
	}
}
listen {
	ipv6addr = ::
	port = 0
	type = acct
	limit {
	}
}
authorize {
	filter_username
	preprocess
	chap
	mschap
	digest
	suffix
	eap {
		ok = return
	}
	files
	-sql
	-ldap
	expiration
	logintime
	pap
```

```
	Autz-Type New-TLS-Connection {
		ok
	}
}
authenticate {
	Auth-Type PAP {
		pap
	}
	Auth-Type CHAP {
		chap
	}
	Auth-Type MS-CHAP {
		mschap
	}
	mschap
	digest
	eap
}
preacct {
	preprocess
	acct_unique
	suffix
	files
}
accounting {
	detail
	unix
	-sql
	exec
	attr_filter.accounting_response
}
session {
}
post-auth {
	if (session-state:User-Name && reply:User-Name && request:User-Name && (reply:User-Name ==
request:User-Name)) {
		update reply {
			&User-Name !* ANY
		}
	}
```

```
	update {
		&reply: += &session-state:
	}
	-sql
	exec
	remove_reply_message_if_eap
	Post-Auth-Type REJECT {
		-sql
		attr_filter.access_reject
		eap
		remove_reply_message_if_eap
	}
	Post-Auth-Type Challenge {
	}
	Post-Auth-Type Client-Lost {
	}
	if (EAP-Key-Name && &reply:EAP-Session-Id) {
		update reply {
			&EAP-Key-Name := &reply:EAP-Session-Id
		}
	}
}
pre-proxy {
}
post-proxy {
	eap
}
}
```

```
# file: sites-available/inner-tunnel
server inner-tunnel {
listen {
       ipaddr = 127.0.0.1
       port = 18120
       type = auth
}
authorize {
	filter_username
```

```
	chap
	mschap
	suffix
	update control {
		&Proxy-To-Realm := LOCAL
	}
	eap {
		ok = return
	}
	files
	-sql
	-ldap
	expiration
	logintime
	pap
}
authenticate {
	Auth-Type PAP {
		pap
	}
	Auth-Type CHAP {
		chap
	}
	Auth-Type MS-CHAP {
		mschap
	}
	mschap
	eap
}
session {
	radutmp
}
post-auth {
	-sql
	if (0) {
		update reply {
			User-Name !* ANY
			Message-Authenticator !* ANY
			EAP-Message !* ANY
			Proxy-State !* ANY
```

```
        MS-MPPE-Encryption-Types !* ANY
        MS-MPPE-Encryption-Policy !* ANY
        MS-MPPE-Send-Key !* ANY
        MS-MPPE-Recv-Key !* ANY
    }
    update {
        &outer.session-state: += &reply:
    }
}
Post-Auth-Type REJECT {
    -sql
    attr_filter.access_reject
    update outer.session-state {
        &Module-Failure-Message := &request:Module-Failure-Message
    }
}
}
pre-proxy {
}
post-proxy {
eap
}
} # inner-tunnel server block
```

```
# file: sites-available/tls
listen {
ipaddr = *
port = 2083
type = auth+acct
proto = tcp
virtual_server = default
clients = radsec
limit {
      max_connections = 16
      lifetime = 0
      idle_timeout = 30
}
tls {
```

```
        private_key_password = whatever
        private_key_file = /etc/ssl/private/ssl-cert-snakeoil.key
        certificate_file = /etc/ssl/certs/ssl-cert-snakeoil.pem
        ca_file = /etc/ssl/certs/ca-certificates.crt
        fragment_size = 8192
        ca_path = ${cadir}
        ca_path_reload_interval = 3600
        cipher_list = "DEFAULT"
        cipher_server_preference = no
        tls_min_version = "1.2"
        tls_max_version = "1.3"
        ecdh_curve = ""
        cache {
             enable = no
             lifetime = 24 # hours
        }
        require_client_cert = yes
        verify {
        }
    }
}
clients radsec {
    client 127.0.0.1 {
        ipaddr = 127.0.0.1
        proto = tls
        secret = radsec
    }
}
home_server tls {
    ipaddr = 127.0.0.1
    port = 2083
    type = auth
    secret = radsec
    proto = tcp
    status_check = none
    tls {
        private_key_password = whatever
        private_key_file = /etc/ssl/private/ssl-cert-snakeoil.key
        certificate_file = /etc/ssl/certs/ssl-cert-snakeoil.pem
        ca_file = /etc/ssl/certs/ca-certificates.crt
```

```
    fragment_size = 8192
    ca_path = ${cadir}
    cipher_list = "DEFAULT"
    connect_timeout = 30
  }
}
home_server_pool tls {
    type = fail-over
    home_server = tls
}
realm tls {
      auth_pool = tls
}
```

-end

Revision #12
Created 24 June 2024 20:37:12 by bluecrow76
Updated 13 January 2026 23:14:39 by bluecrow76