

# radsecproxy

[radsecproxy Github Project Page](#)

[radsecproxy.conf man page](#)

[Configuration Example](#)

## Configuration Options

```
# Note that some block option values may reference a block by name, in which case the block name must be
previously defined. Hence the order of the blocks may be significant.

# Recommended block order:

#  tls
#  rewrite
#  client
#  server
#  realm


# The rewrite actions are performed in this sequence:
# 1. RemoveAttribute (or WhitelistAttribute)
# 2. ModifyAttribute
# 3. SupplementAttribute
# 4. AddAttribute

rewrite name {
    AddAttribute attribute:value
    AddVendorAttribute vendor:subattribute:value
    SupplementAttribute attribute:value
    SupplementVendorAttribute vendor:subattribute:value
    ModifyAttribute attribute:/regex/replace/
    ModifyVendorAttribute vendor:subattribute:/regex/replace/
    RemoveAttribute attribute
    RemoveVendorAttribute vendor[:subattribute]
    WhitelistMode (on|off)
```

```
    WhitelistAttribute attribute
    WhitelistVendorAttribute vendor[:subattribute]
}
```

```
tls name {
    CACertificateFile file
    CACertificatePath path
    CertificateFile file
    CertificateKeyFile file
    CertificateKeyPassword password
    PolicyOID oid
    CRLCheck (on|off)
    CacheExpiry seconds
}
```

```
client (name|fqdn|(address[/length])) {
    Host (fqdn|(address[/length])) # multiple lines allowed
    IPv4Only (on|off)
    IPv6Only (on|off)
    Type type (UDP|TCP|TLS|DTLS)
    Secret secret
    TLS tls
    CertificateNameCheck (on|off)
    matchCertificateAttribute ( CN | SubjectAltName:URI | SubjectAltName:DNS ) :/regexp/
    MatchCertificateAttribute SubjectAltName:IP:address
    DuplicateInterval seconds
    AddTTL 1-255
    TCPKeepalive (on|off)
    FticksVISCOUNTRY cc
    FticksVISINST institution
    RewriteIn rewrite
    RewriteOut rewrite
    RewriteAttribute User-Name:/regex/replace/
}
```

```
server (name|((fqdn|address)[:port])) {
    Host (fqdn|address)[:port]
    Port port
    DynamicLookupCommand command
    StatusServer (on|off|minimal|auto)
```

```

    ⌈RetryCount count
    ⌈RetryInterval interval
    ⌈RewriteOut rewrite
    ⌈RewriteIn rewrite
    ⌈LoopPrevention (on|off)
    ⌈IPv4Only (on|off)
    ⌈IPv6Only (on|off)
    ⌈Type type
    ⌈Secret secret
    ⌈TLS tls
    ⌈CertificateNameCheck (on|off)
    ⌈matchCertificateAttribute ( CN | SubjectAltName:URI | SubjectAltName:DNS ) :/regex/
    ⌈MatchCertificateAttribute SubjectAltName:IP:address
    ⌈AddTTL 1-255
    ⌈TCPKeepalive (on|off)
}

realm (*|realm|/regex/) {
    ⌈Server server
    ⌈AccountingServer server
    ⌈AccountingResponse (on|off)
    ⌈ReplyMessage message
}

```

# Configuration framework

Create the folders and files first:

```

mkdir /etc/radsecproxy.d

touch /etc/radsecproxy.d/tls.conf
touch /etc/radsecproxy.d/rewrites.conf
touch /etc/radsecproxy.d/clients.conf
touch /etc/radsecproxy.d/servers.conf
touch /etc/radsecproxy.d/realms.conf

```

```
chown -R radsecproxy:radsecproxy /etc/radsecproxy.*
```

```
chmod 770 /etc/radsecproxy.d
```

```
chmod 660 /etc/radsecproxy.conf /etc/radsecproxy.d/*.conf
```

Now populate the files as needed:

```
# /etc/radsecproxy.conf
```

```
IPv4Only          on
```

```
ListenUDP         *:1812
```

```
ListenUDP         *:1813
```

```
ListenTLS         *:2083
```

```
# For testing later reduce to 3
```

```
LogLevel          4
```

```
#LogDestination   file:///var/log/radsecproxy.log
```

```
LogDestination    x-syslog:///
```

```
LoopPrevention    on
```

```
Include /etc/radsecproxy.d/tls.conf
```

```
Include /etc/radsecproxy.d/rewrites.conf
```

```
Include /etc/radsecproxy.d/clients.conf
```

```
Include /etc/radsecproxy.d/servers.conf
```

```
Include /etc/radsecproxy.d/realms.conf
```

```
# file:tls.conf
```

```
tls default {
```

```
    CACertificateFile /etc/radsecproxy.d/certs/trusted-roots.crt
```

```
    CertificateFile   /etc/radsecproxy.d/certs/certificate.crt
```

```
    CertificateKeyFile /etc/radsecproxy.d/certs/certificate.private.key
```

```
    CRLCheck off
```

```
    #CacheExpiry 3600
```

```
    #policyOID      1.3.6.1.5.5.7.3.2
```

```
}
```

```
# file:rewrites.conf
```

```
# examples of possible rewrites
```

```
# @azuremfa -> @mydomain.com : for forcing Azure MFA
```

```
rewrite azuremfa {  
    modifyAttribute 1:/^(.*)@azure$/\1@mydomain.com/  
    modifyAttribute 1:/^(.*)@azuremfa$/\1@mydomain.com/  
}
```

```
# @azurenomfa -> @mydomain.com : for forcing logins that don't require MFA
```

```
rewrite azurenomfa {  
[]modifyAttribute 1:/^(.*)@azurenomfa$/\1@mydomain.com/  
}
```

```
# @router -> @mydomain.com : for logging into devices with MFA required
```

```
rewrite azuremfa-router {  
    modifyAttribute 1:/^(.*)@router$/\1@mydomain.com/  
}
```

```
# @duo -> @mydomain.com : for forcing DUO MFA
```

```
rewrite duomfa {  
    modifyAttribute 1:/^(.*)@duo$/\1@mydomain.com/  
[]modifyAttribute 1:/^(.*)@duomfa$/\1@mydomain.com/  
}
```

```
# @oldBusinessName -> [norealml] : for forcing login to legacy Active Directory
```

```
rewrite oldBusinessName {  
[]modifyAttribute 1:/^(.*)@oldBusinessName$/\1/  
}
```

```
# adding a radsecproxy identifying attribute for incoming client requests
```

```
rewrite tagrsp1 {  
[]RemoveVendorAttribute 14988:11  
[]AddVendorAttribute 14988:11:"rsp1"  
}
```

```
# [username]@legacyRadius -> [username] : for forcing logins via a legacy radius server
```

```
# logins should have been presented to us a username@legacyRadius
```

```
# we need to pass them the legacy radius server with no @legacyRadius
```

```
rewrite legacyRadius {  
    modifyAttribute 1:/^(.*)@legacyRadius$/\1/  
}
```

```
# file:clients.conf
```

```
client office1 {  
    host 1.1.1.1  
    host 2.2.2.2  
    host 10.1.2.1  
    #rewriteIn tagrsp1  
    secret aBetterPasswordThanThis  
    type udp  
}
```

```
client office2 {  
    host 3.3.3.3  
    host 10.3.2.1  
    #rewriteIn tagrsp1  
    secret aBetterPasswordThanThis  
    type udp  
}
```

```
client catchallIPv4UDP {  
    host 0.0.0.0/0  
    #rewriteIn tagrsp1  
    secret aBetterPasswordThanThis  
    type udp  
}
```

```
client catchallIPv4TLS {  
    host 0.0.0.0/0  
    #rewriteIn tagrsp1  
    type TLS  
}
```

```
# file:servers.conf
```

```
# Azure NPS 001 - No MFA  
server azureNPS01 {
```

```

    host 192.168.1.11:1812
    type udp
    secret aBetterPasswordThanThis
}
RetryCount 0
    RetryInterval 3
    RewriteOut azurenomfa
}

# Azure NPS 002 - MFA required
server azureNPS02MFA {
    host 192.168.1.12:1812
    type udp
    secret aBetterPasswordThanThis
}
RetryCount 0
    RetryInterval 30
    RewriteOut azuremfa
}

# Azure NPS 002 - MFA Required - with router specific rewrite
server azureNPS02MFA-router {
    host 192.168.1.12:1812
    type udp
    secret aBetterPasswordThanThis
}
RetryCount 0
    RetryInterval 30
    RewriteOut azuremfa-router
}

# DUO MFA
server duoauthproxy {
    host 127.0.0.1:31812
    type udp
    secret aBetterPasswordThanThis
}
RetryCount 0
    RetryInterval 30
    RewriteOut duomfa
}

# Legacy Active Directory
server legacyAD {

```

```
    host 192.168.2.11:1812
    type udp
    secret aBetterPasswordThanThis
}
RetryCount 0
    RetryInterval 30
    RewriteOut legacyRadius
}

# Legacy Radius Server
server legacyRadius01 {
    host 8.9.10.11:1812
    type udp
    secret aBetterPasswordThanThis
}
RetryCount 0
    RetryInterval 5
    RewriteOut legacyRadius
}
```

```
# file:realms.conf
```

```
# Force Azure MFA
```

```
realm azure {
    server azureNPS02MFA
    AccountingResponse on
}
```

```
# Force Azure MFA
```

```
realm azuremfa {
    server azureNPS02MFA
    AccountingResponse on
}
```

```
# Force Azure No MFA
```

```
realm azurenomfa {
    server azureNPS01
    AccountingResponse on
}
```

```
# Force DUO MFA
```

```
realm duo {
```



```
server duoauthproxy
AccountingResponse on
}
```

# Force DUO MFA

```
realm duomfa {
server duoauthproxy
AccountingResponse on
}
```

# Force legacy Active Directory

```
realm mydomain.com {
server legacyRadius
AccountingResponse on
}
```

# Force router management login - service against legacy radius server

```
realm router {
server legacyRadius
AccountingResponse on
}
```

# For legacy Active Directory

```
realm old {
server legacyRadius
AccountingResponse on
}
```

# All other realms

```
realm * {
server legacyRadius
AccountingResponse on
# ReplyMessage "Contact tech support..."
}
```

# All other realms - deny all other requests

# Obviously you can only have one wildcard matcher, this is just another option

```
realm * {
ReplyMessage "Radius request denied by proxy. Contact tech support."
}
```

---

Revision #8

Created 13 April 2022 19:44:03 by bluecrow76

Updated 2 December 2023 07:20:45 by bluecrow76