Installation

Replication of production setup

Here, we replicate the relevant configuration already present on server.lastschl.av as a starting point. The test virtual machine will have two network interfaces, one serving as uplink on a 192.168.10.0/24 network (IP 192.168.10.52), and one to connect to the NASes/supplicants on a 192.168.9.0/24 network (IP 192.168.9.1). The FQDN will be server.test.av.

Base virtual machine preparation

Imported Last School's Debian9 VM template "Debian9-base.ova" into Virtual Box as Debian9-base_8021x, re-initializing all MAC addresses. The description for this virtual machine template is:

```
Debian 9 amd64 installation
- Hostname:
  debian9-base
- User accounts {username password}:
  ls last
  root last
- Partitioning:
  --- Physical:
  ------ 1GB RAID boot flag
  ------ 29GB RAID
  --- RAID:
  ------ md0: ext3 /boot
  ------ md1: LVM - part of volume group debian9-base
  --- LVM (VG/LV):
  ------ debian9-base/root: 18.6GB ext4 /
  ------ debian9-base/swap: 3.72GB swap area
- Up to date as of 2017-09-27
- sources.list includes:
  Sections: main contrib non-free
  Additional repository: backports
- Apt-cacher configured as per Last School site (Proxy credentials will need to be entered in /etc/apt/apt.conf.d/02proxy by user)
- SSH access installed and enabled
- Gnome and Firefox configured to auto-detect proxy settings
- Extra software installed:
  vlc gimp emacs fonts-indic tcpdump iperf exfat-utils wireshark
- One network interface as bridged adapter, cable connected.
```

Added a second ethernet adapter in settings, connected to "Bridged adapter", re-initialized its MAC address

Increased the allocated CPUs to 2

The host computer has two network interfaces, one connected to a network uplink and another connected to a couple of NASes. Each VirtualBox virtual interface is bridged to a different physical adapter. Network configuration is now as follows (interface name seen in guest OS - Adapter name in VirtualBox settings - Adapter "Attached to" setting in VirtualBox settings - Physical interface bridged to):

```
enp0s3 - Adapter 1 - Bridged adapter - physical interface connected to uplink
enp0s8 - Adapter 2 - Bridged adapter - physical interface connected to NASes
```

Booted the VM, logged in to the GUI, connected using DHCP with network manager

In terminal:
```
rm /etc/apt/apt.conf.d/02proxy
apt-get update
apt-get upgrade
```

Rebooted the virtual machine

Set strong passwords for ls and root users

Installed my ssh public key in root's .ssh/authorized_keys file.
Installation of relevant services:

Shorewall (based on LASTSCHL-207):

```
apt-get install shorewall
apt-get install ipset
mv /etc/shorewall{,-orig}
mkdir /etc/shorewall
```

Configuration:
for i in `ls`; do echo "========= $i ========="; cat $i | grep -v "^#" | grep -v "^"; echo "========= $i ========="; echo ""; done

===== interfaces =====
net    enp0s3         detect      tcpflags,dhcp,nosmurfs,routefilter,logmartians
wifi   enp0s8                   detect           tcpflags,nosmurfs,routefilter,logmartians

===== masq =====
enp0s3                                           192.168.9.0/24

===== policy =====
$FW    net                REJECT                INFO(uid)
$FW    wifi                ACCEPT                INFO(uid)
net    all                DROP                  INFO
all    all                REJECT                info

===== rules =====
Invalid(DROP)                 net                        all
ACCEPT:INFO(uid)     net                        $FW                        tcp     22
ACCEPT:INFO(uid)     net                        $FW                        udp     123
ACCEPT:INFO(uid)     net                        $FW                        icmp
ACCEPT:INFO(uid)     $FW                        net                        tcp     465,587,995,993
ACCEPT:INFO(uid)     $FW                        net                        udp     53,123
ACCEPT:INFO(uid)     $FW                        net                        icmp
ACCEPT:INFO(uid)     $FW                        net                        tcp
-               -               root
ACCEPT:INFO(uid)     $FW                        net                        udp
-               -               root
ACCEPT:INFO(uid)     $FW                        net                        icmp
-               -               root
ACCEPT:INFO(uid)     $FW                        net                        tcp
-               -               _apt
ACCEPT:INFO(uid)     $FW                        net                        udp
-               -               _apt
ACCEPT:INFO(uid)     $FW                        net                        icmp
-               -               _apt

In /etc/default/shorewall, set
startup=1

In /etc/default/shorewall, set

root@debian9-base:~# cat /etc/rsyslog.d/40-shorewall.conf
:msg, contains, "Shorewall:" /var/log/shorewall
& stop

root@debian9-base:~# cat /etc/logrotate.d/shorewall
/var/log/shorewall-init.log {
    weekly
    rotate 108
    compress
    nomissingok
    create 0640 root adm
}
/var/log/shorewall{
    rotate 731
daily
nomissingok
notifempty
delaycompress
compress
dateext
postrotate
    reload rsyslog >/dev/null 2>&1 || true
endscript
}
root@debian9-base:~# cat /etc/logrotate.d/rsyslog
/var/log/syslog
/var/log/auth.log{
    rotate 731
daily
dateext
nomissingok
notifempty
delaycompress
compress
postrotate
    invoke-rc.d rsyslog rotate > /dev/null
endscript
}
/var/log/mail.info
/var/log/mail.warn
/var/log/mail.err
/var/log/mail.log
/var/log/daemon.log
/var/log/kern.log
/var/log/user.log
/var/log/lpr.log
/var/log/cron.log
/var/log/debug
/var/log/messages{
    rotate 4
weekly
missingok
notifempty
delaycompress
compress
sharedscripts
postrotate
    invoke-rc.d rsyslog rotate > /dev/null
endscript
}
root@debian9-base:~# cat /etc/logrotate.conf
# see "man logrotate" for details
# rotate log files weekly
weekly
# keep 4 weeks worth of backlogs
rotate 4
# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own wtmp, or btmp -- we'll rotate them here
/var/log/wtmp {
  nomissingok
  monthly
  create 0664 root utmp
  rotate 24
}

/var/log/btmp {
  nomissingok
  monthly
  create 0660 root utmp
  rotate 24
}

# system-specific logs may be configured here

systemctl enable shorewall.service

Configure network and DHCP (based on LASTSCHL-212):

systemctl disable network-manager.service
systemctl disable NetworkManager.service
unlink /etc/resolv.conf
echo nameserver 192.168.10.1 > /etc/resolv.conf
mkdir /etc/ltsp
Configure DNS (based on LASTSCHL-211):

```
apt-get install dnsmasq
touch /var/log/dnsmasq
chmod 640 /var/log/dnsmasq
```

```
root@debian9-base:~# cat /etc/dnsmasq.conf | grep -v "^#" | grep -v "^$"
strict-order
interface=enp0s8
expand-hosts
domain=test.av
log-queries
log-facility=/var/log/dnsmasq
```

```
root@debian9-base:~# cat /etc/logrotate.d/dnsmasq
/var/log/dnsmasq
{
    rotate 731
daily
    nomissingok
    notifempty
delaycompress
    compress
dateext
    postrotate
    reload rsyslog >/dev/null 2>&1 || true
   endscript
}
```

```
root@debian9-base:~# cat /etc/hostname
server.test.av
```

```
root@debian9-base:~# cat /etc/hosts
127.0.0.1 localhost
192.168.9.1 test.av
192.168.9.1 server.test.av server

# The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

New stuff

Now that we have a working setup similar to the production one, we will modify it to implement the new solution.

Download the latest version of the attached `shwl_add_shwl_del_sl_pmu` archive and extract it somewhere.

Shorewall

Add to `/etc/shorewall/hosts`:

```
wifi1 enp0s8:dynamic
```

Modify `/etc/shorewall/policy:`
Add to /etc/shorewall/zones:

```
... 
wifi1: wifi ipv4 dynamic_shared
```

In /etc/shorewall/shorewall.conf set:

```
SAVE_IPSETS=Yes
```

Add to /etc/shorewall/rules (replace IP addresses with actual IP address of NASes):

```
# At the top of the file:
?SECTION ALL
# Allow the server and NASes to talk RADIUS and HTTP (web interface)
ACCEPT wifi:192.168.9.2,192.168.9.3,192.168.9.4 
$FW tcp -  80
ACCEPT $FW wifi:192.168.9.2,192.168.9.3,192.168.9.4 tcp  80 -
ACCEPT wifi:192.168.9.2,192.168.9.3,192.168.9.4 
$FW udp 1812 -
# But, reject anything else to and from any other device part of the 192.168.9.0/24 network that is not part of 
# any dynamic zone
REJECT wifi all - - - 
REJECT all - - -
wifi - - - 
?SECTION NEW
# At the end of the file:
ACCEPT:INFO(uid) wifi:192.168.9.2,192.168.9.3,192.168.9.4 $FW udp 1812
```

**FreeRADIUS**

```
apt-get install freeradius
systemctl enable freeradius.service
```

Modify /etc/freeradius/3.0/mods-available/eap:

```
comment the following:
```
modify the 'default_eap_type' directive under section 'eap' to be:

default_eap_type = peap

and the 'default_eap_type' directive under section 'ttls' to be:

default_eap_type = mschapv2

Modify /etc/freeradius/3.0/sites-available/default, comment the following lines (see comments included in the code block):
# All the listen sections except the IPv4 version with "type = auth"

```bash
listen {
    ipaddr = *
    port = 0
    type = acct
    limit {
    }
}

type = auth
listen {
    ipv6addr = ::       # any. ::1 == localhost
    port = 0
    limit {
        max_connections = 16
        lifetime = 0
        idle_timeout = 30
    }
}
listen {
    ipv6addr = ::
    port = 0
    type = acct
    limit {
    }
}

# In the authorize section:
chap
mschap
digest
files
-ldap
pap

# In the authenticate section:
Auth-Type PAP {
    pap
}
Auth-Type CHAP {
    chap
}
Auth-Type MS-CHAP {
    mschap
}
mschap
digest

Uncomment the following line in the 'authorize' section:

```bash
auth_log
```

Add the following line at the end of the 'post-auth' section and at the beginning of the Post-Auth-Type REJECT section:

```bash
reply_log
```
update reply {
    Session-Timeout := 3600
    Termination-Action := 1
}

Modify /etc/freeradius/3.0/sites-available/inner-tunnel, comment the following lines:

```plaintext
# The whole listen section
listen {
    ipaddr = 127.0.0.1
    port = 18120
    type = auth
}
# In the authorize section:
chap
mschap
files
-ldap
# In the authenticate section:
Auth-Type PAP {
    pap
}
Auth-Type CHAP {
    chap
}
Auth-Type MS-CHAP {
    mschap
}
```

Add the following line after 'filter_username' and before 'suffix' in the 'authorize' section:

```plaintext
auth_log
```

Add the following line at the end of the 'post-auth' section and at the beginning of the Post-Auth-Type REJECT section:

```plaintext
reply_log
```

Add the following section just below the Post-Auth-Type REJECT section:

```plaintext
Post-Auth-Type CHALLENGE {
    reply_log
}
```

Modify /etc/freeradius/3.0/radiusd.conf, set (in the 'log' section):

```plaintext
auth = yes
```

Modify /etc/freeradius/3.0/clients.conf, comment the 'client localhost' and 'client localhost_ipv6' section and add (replace with actual IP addresses of NASes):

```plaintext
```
client wifi-ap1 {
    ipaddr = 192.168.9.2
    secret = password # Replace with an actual password
}
client wifi-ap2 {
    ipaddr = 192.168.9.3
    secret = password # Replace with an actual password
}
client wifi-ap3 {
    ipaddr = 192.168.9.4
    secret = password # Replace with an actual password
}

Modify /etc/logrotate.d/freeradius, modify the following options as follows ('dateext' option needs to be added):

rotate 732
nomissingok
dateext

rm /var/log/freeradius/radius.log
rm /var/log/freeradius/radwtmp
chmod o-rwx /var/log/freeradius
chown freerad:freerad /var/log/freeradius
chmod o-rwx /etc/freeradius

It has been observed that radius.log comes with world-readable permissions upon installation of the package, deleting it causes FreeRADIUS to re-create it, and it gets re-created with more secure permissions. /etc/freeradius also comes with the executable bit set for all users, which makes it easier for sensitive information contained within to be world-readable in case the permissions of an individual file are not set restrictive enough (as was, by default, the case with the file containing the encryption passwords for the certificates). Could not find any information on the net on whether there is a good reason for the executable bit being set, so, decided it is safer to remove it.

Certificates

Modify /etc/freeradius/3.0/certs/server.cnf, set the following settings:

...  
[ CA_default ]
...  
default_days = 732
...  
[ req ]
...  
input_password = password # Replace with an actual password
output_password = password # Replace with an actual password, should be same as input_password
...  
[certificate_authority]
countryName = IN
stateOrProvinceName = Tamil Nadu
localityName = Auroville
organizationName = Test
emailAddress = admin@test.av
commonName = "Test Certificate Authority"
...  

Modify /etc/freeradius/3.0/certs/ca.cnf, set the following settings:
... [ CA_default ]
... default_days = 732
... crlDistributionPoints = URI:http://server.test.av/test_ca.crl # This URL will not actually be implemented at the moment, but choose a URL where it is possible to in future make the file available

[ req ]
... input_password = password # Replace with an actual password, different from the one in server.cnf
output_password = password # Replace with an actual password, should be same as input_password

[server]
countryName = IN
stateOrProvinceName = Tamil Nadu
localityName = Auroville
organizationName = Test
emailAddress = admin@test.av
commonName = "Test Server Certificate"

[v1_ca]
... crlDistributionPoints = URI:http://server.test.av/test_ca.crl

Modify /etc/freeradius/3.0/certs/xpextensions, set the following settings:

... [ xpclient_ext]
... crlDistributionPoints = URI:http://server.test.av/test_ca.crl # This URL will not actually be implemented at the moment, but choose a URL where it is possible to in future make the file available
...

[ xpserver_ext]
... crlDistributionPoints = URI:http://server.test.av/test_ca.crl # This URL will not actually be implemented at the moment, but choose a URL where it is possible to in future make the file available
...

Modify /etc/freeradius/3.0/certs, set the following settings:

cd /etc/freeradius/3.0/certs
rm -f *.pem *.der *.csr *.crt *.key *.p12 serial* index.txt* # This step is probably not needed,
make ca.pem
make ca.der
make server.pem
make server.csr
chown freerad:freerad *
chmod o-rwx *
rm bootstrap
rm passwords.mk
# Delete all other files in the folder except: server.cnf, ca.cnf, xpextensions, Makefile, README, dh, ca.pem, server.pem, server.key

Modify /etc/freeradius/3.0/mods-available/eap, modify the following directives under section 'tls-config tls-common' to be:

private_key_password = password # Replace password with the password chosen previously in server.cnf
private_key_file = /etc/freeradius/3.0/certs/server.pem
......
 certificate_file = /etc/freeradius/3.0/certs/server.pem
 ....
 ca_file = /etc/freeradius/3.0/certs/ca.pem
MySQL

```
apt-get install mysql-server freeradius-mysql
mysql -uroot
  CREATE DATABASE radius;
  exit
mysql -uroot radius < /etc/freeradius/3.0/mods-config/sql/main/mysql/schema.sql
```

Edit /etc/freeradius/3.0/mods-config/sql/main/mysql/setup.sql. Modify the following lines:

```
CREATE USER 'radius'@'localhost';
SET PASSWORD FOR 'radius'@'localhost' = PASSWORD('radpass');
```

to

```
CREATE USER 'freerad'@'localhost' IDENTIFIED VIA unix_socket;
```

and update the username 'radius' to be 'freerad' wherever else it is mentioned in the file.

```
mysql -uroot radius < /etc/freeradius/3.0/mods-config/sql/main/mysql/setup.sql
```

cd /etc/freeradius/3.0/mods-enabled
ln -s ../mods-available/sql sql

In /etc/freeradius/3.0/mods-enabled/sql, set the following options:

```
driver = "rlm_sql_mysql"
dialect = "mysql"
server = "localhost"
port = 3306
login = "freerad"
pattern = "*"
radius_db = "radius"
logfile = ${logdir}/sqllog.sql
```

Modify /etc/freeradius/3.0/sites-enabled/inner-tunnel, find the following line under authorize, post-auth and Post-Auth-Type REJECT sections

```
-sql
```

modify it to

```
sql
```

Modify /etc/freeradius/3.0/sites-enabled/default, find the following line under authorize, post-auth and Post-Auth-Type REJECT sections

```
-sql
```

In the post-auth and Post-Auth-Type REJECT sections, modify it to

```
sql
```

In the authorize section, comment it out.

**Python module / script_launcher.py script**
apt-get install libpython2.7-dev # It is not fully sure whether this package is needed

cd /etc/freeradius/3.0/mods-enabled
ln -s ../mods-available/python python

Put the following in it:

```
/etc/freeradius/3.0/mods-enabled/python

# Make sure the PYTHONPATH environmental variable contains the
# directory(s) for the modules listed below.
#
# Uncomment any func_* which are included in your module. If
# rlm_python is called for a section which does not have
# a function defined, it will return NOOP.
#
python {
    module = script_launcher

    python_path = ${modconfdir}/${.:name}:${modconfdir}/usr/lib/python2.7

    mod_post_auth = ${.module}
    func_post_auth = post_auth
}
```

Modify /etc/freeradius/3.0/sites-enabled/inner-tunnel:

```
/etc/freeradius/3.0/sites-enabled/inner-tunnel

... # Add this line just after 'sql' in the 'post-auth' section
python ...
```

Modify /etc/freeradius/3.0/mods-available/eap, modify the 'copy_request_to_tunnel' directive under both sections 'peap' and 'ttls' to be:

```
copy_request_to_tunnel = yes
```

Place the script_launcher.py script from the shwl_add_shwl_del_sl_pmu archive at /etc/freeradius/3.0/mods-config/python/script_launcher.py

```
chown freerad:freerad /etc/freeradius/3.0/mods-config/python/script_launcher.py
chmod 640 /etc/freeradius/3.0/mods-config/python/script_launcher.py
```

```
sudo
apt-get install sudo
```

Create /etc/sudoers.d/shwl_add_shwl_del_pmu, permissions 640 root:root, with:

```
freenrad ALL=(root:root) NOPASSWD:/:sbin/shorewall,:/usr/bin/arp-scan
```
shwl_add / shwl_del scripts

Prerequisites from above steps: sudo, FreeRADIUS python module / script_launcher.py script, shorewall, FreeRADIUS MySQL

apt-get install arp-scan
# Install the shwl_*.sh scripts from the shwl_add_shwl_del_sl_pmu archive in /usr/local/sbin/
chmod 750 /usr/local/sbin/shwl_*
mkdir /var/local/shwl_add
chmod 700 /var/local/shwl_add
chmod a-s /var/local/shwl_add

Add the following line to freerad’s crontab

*/1 * * * * /usr/local/sbin/shwl_del.sh

Settings in /usr/local/sbin/shwl_add.sh (at top of file):

....
# Settings
cfg_shwl_zone="wifi1" # Shorewall dynamic zone to which client devices' IP addresses need to be added
cfg_shwl_retry_delay=2 # Number of seconds to wait, in case of failure in adding IP to shorewall dynamic zone, before attempting second time
cfg_file_location="/var/local/shwl_add" # Folder where runtime information will be stored
cfg_file_location_owner_user="freerad" # User by which above folder should be owned
cfg_file_location_owner_group="freerad" # Group by which above folder should be owned
cfg_ip_srch_iface="enp0s8" # Network interface on which to scan for devices
cfg_ip_srch_initial_delay=1 # How many seconds to wait before first attempt at scanning
cfg_ip_srch_retry_delay=2 # How many seconds to wait in between further attempts at scanning
cfg_ip_srch_max_attempts=15 # Maximum number of attempts at scanning before giving up
cfg_mysql_user="freerad" # MySQL username
cfg_mysql_db="shwl_add_shwl_del_pmu" # MySQL database name where to log events
cfg_mysql_log_table="event_log" # Table in MySQL database where to log events
....

Settings in /usr/local/sbin/shwl_del.sh (at top of file):

....
# Settings
cfg_ip_match_pattern="192.168." # Pattern to match all IP addresses that might be in the shorewall dynamic zone
cfg_session_expiry_timeout=3720 # Session duration (should be slightly longer than Session-Timeout attribute specified in FreeRADIUS)
cfg_shwl_zone="wifi1" # Shorewall dynamic zone containing clients' IP addresses
cfg_file_location="/var/local/shwl_add" # Folder where runtime information is stored
cfg_file_location_owner_user="freerad" # User by which above folder should be owned
cfg_file_location_owner_group="freerad" # Group by which above folder should be owned
cfg_mysql_user="freerad" # MySQL username
cfg_mysql_db="shwl_add_shwl_del_pmu" # MySQL database name where to log events
cfg_mysql_log_table="event_log" # Table in MySQL database where to log events
....

MySQL
pam_to_mysql_update.sh script

Prerequisites from above: sudo, FreeRADIUS MySQL, shwl_add / shwl_del scripts MySQL

```
apt-get install libpam-script sshpass
mkdir /usr/share/libpam-script/pam-script.d/pam_to_mysql_update
cd /usr/share/libpam-script/pam-script.d/pam_to_mysql_update
# Install the pam_to_mysql_update.sh script from the shwl_add_shwl_del_sl_pmu archive in here
in -s pam_to_mysql_update.sh pam_script_auth
in -s pam_to_mysql_update.sh pam_script_passwd
mysql -uroot
  GRANT ALL on radius.radcheck TO 'freerad'@'localhost';
exit
pam-auth-update # And, uncheck the box for "Support for authentication by external scripts"
```

Add the following line at the end of `/etc/pam.d/common-auth`:

```
/etc/pam.d/common-auth

....
auth        required                        pam_script.so onerr=fail dir=/usr/share/libpam-script/pam-script.d
           /pam_to_mysql_update/
```

Add the following line at the end of `/etc/pam.d/common-password`:

```
/etc/pam.d/common-password

....
password    required                        pam_script.so onerr=fail dir=/usr/share/libpam-script/pam-
            script.d/pam_to_mysql_update/
```

Settings in `/usr/share/libpam-script/pam-script.d/pam_to_mysql_update/pam_to_mysql_update.sh` (at top of file):

```
....
# Settings
cfg_mysql_user="freerad" # MySQL username
cfg_mysql_user_db="radius" # MySQL database name where to update passwords
cfg_mysql_log_db="shwl_add_shwl_del_pmu" # MySQL database name where to log events
cfg_mysql_user_table="radcheck" # Table in MySQL database where to update passwords
cfg_mysql_log_table="event_log" # Table in MySQL database where to log events
cfg_verbose=0 # Print verbose messages on stdout
....
```

Managing users

The below procedures also create/change password for/delete system users as well as users for FreeRADIUS.
Adding users
Replace 'user' with the desired username.

```
mysql -uroot
    use radius;
    INSERT INTO radcheck VALUES ('','user','NT-Password','::','');
exit
adduser user # When prompted for "Current password:“, ignore and press enter
```

Changing password
It is sufficient to use standard utilities such as 'passwd', the password will be updated in the MySQL database as well. See the 'pam_to_mysql_update.sh' section in the 'Overview' page for utilities with which this has been tested. In case prompted with "Current password:" (exactly as written here) it is sufficient to ignore and press enter. Commands that expire or disable a user's system user account (or password) without deleting it (such as passwd -l) will not cause the credentials in the MySQL database to be disabled, thus it is necessary to take care (perhaps with a site specific lock user script) to also invalidate the same.

Deleting users
Replace 'user' with the username to be deleted.

```
mysql -uroot
    use radius;
    DELETE FROM radcheck WHERE username='user';
exit
deluser user
```

WiFi access point (NAS) configuration

Settings to be configured
These are the settings that usually need to be configured, on dual-band routers it might be necessary to configure some of the settings twice, once under the settings for the 2.4GHz SSID and once for the 5GHz SSID:

- SSID - SSID of choice
- Network security type: WPA2 Enterprise
- WPA type: Set to either Auto or WPA2
- WPA encryption: Set to either Auto or AES
- RADIUS server IP - 192.168.9.1
- RADIUS server port - 1812
- RADIUS server secret/password - Password chosen in clients.conf for this particular NAS
- WPS - Disable (a good NAS should disable it automatically when choosing WPA2 enterprise security, but good to make sure)
- Secure password - Choose a secure password for accessing the NAS web (or other) interface. It is important as it controls access to the wireless security settings, and the web (or other) interface is reachable by supplicants connected to the network.
- Clients isolation - If enabled, prevents connected supplicants from talking to each other/seeing each other's traffic. Can improve security if there is either only one NAS installed or each NAS is in a separate broadcast domain (and there is no other device connected, e.g. through wired network, in the same broadcast domain), at the expense of not allowing connected supplicants to communicate directly with each other (e.g. SSH into each other, etc.)
- IP address - IP address needs to match IP mentioned in clients.conf
- Disable DHCP server
- Some models: Reauthentication period - Specify to something equal to or greater than the Session-Timeout specified in /etc/freeradius/3.0/sites-available/default. Some NASes interpret 0 as disabling re-authentication, and might then also ignore any value mentioned by the FreeRADIUS Session-Timeout / Termination-Action attributes.
- Some models: WPA2/RSN preauthentication - It is suggested to try and disable this feature in case issues are encountered such as mentioned in the "General information" page under Misc. Information point 9.
- Some models: Operation mode - Some NASes have an Operation mode setting, which pre-sets/locks some settings to defaults that are appropriate for different kinds of uses, e.g. "DSL Router", "Wireless Router", "Wireless Access Point". This varies by model, but usually something like "Wireless Access Point" is a good first choice, if available, alternatively "Wireless Router"

TP-Link Archer C20 v4 00000004
In this model, the "Reauthentication period" setting is not available, but the router does honor the timeout specified by the RADIUS server. Operation mode can be set to "Access Point". All other settings should be set as mentioned above. This is a dual band router and some settings need to be set in two places, once for each SSID.
TP-Link TD-W8968 V4 0x00000001

In this model, the "Reauthentication period" setting is available as "Network Re-auth Interval", and the router also does honor the timeout specified by the RADIUS server, and follows whichever is smaller. Setting the setting to zero causes the feature to be disabled and the timeout specified by the RADIUS server to also be ignored. Operation mode can be set to "Wireless Router Mode". All other settings should be set as mentioned above.

TP-Link TL-WR740N v4 00000000

In this model, the "Reauthentication period" setting is not available, and the router does not honor the timeout specified by the RADIUS server. The hostapd daemon running on this router supports the feature and works in a similar way as in the above TP-Link TD-W8968 V4, but no option to configure it is available in the web interface, and hostapd is run with this setting set to 0. Judging by the hostapd source code, it is believed (but not tested) that this means, once authenticated, the router might allow the supplicant to continue being part of the network for up to twelve hours without querying the RADIUS server again. A fix is possible to make hostapd run with its default setting of 3600 seconds, but is out of the scope of this wiki page. After applying the fix, and introducing this router in the network with the above two listed models, the issue mentioned in the "General information" page under Misc. Information point 9 was encountered sometimes. The RSN preauthentication option is also not available in the web interface, but it is believed to be possible to disable RSN pre-authentication with a similar fix. No operation mode setting is available. All other settings should be set as mentioned above.

Supplicant configuration

Linux

1. Copy the ca.pem file generated during certificate generation onto the computer.
2. Select the network's SSID from the list in Network Manager.
3. When asked, enter the following information, then press connect:
   CA certificate: Browse and select the ca.pem file
   Identity: the username
   Password: the password
   Inner authentication: MSCHAPv2 (not "MSCHAPv2 (not EAP)"
   Leave all other fields as they are

Android

1. Copy the ca.pem file generated during certificate generation onto the phone.
2. Open the “Settings” app, go to “Wi-Fi” “Advanced settings” “Install certificates”.
3. Select the ca.pem file.
4. Assign it a name of choice
5. Under “Certificate use” select “WiFi”
6. Once again, open the “Settings” app, go to “Wi-Fi”, and select the network's SSID from the list.
7. When asked, enter the following information, then press connect:
   CA certificate: Select the earlier chosen name when installing the ca.pem file
   Identity: the username
   Password: the password
   Leave all other fields as they are

Windows 10

1. Select the network’s SSID from the list of wireless networks
2. Enter username and password
3. When prompted whether to trust the server, confirm

Mac OS

1. Select the network's SSID from the list of wireless networks
2. Enter username and password
3. When prompted whether to trust the server, confirm

iPhone

1. Select the network’s SSID from the list of wireless networks
2. Enter username and password
3. When prompted whether to trust the server, confirm

Security observations
On Linux and Android supplicants it is required to install the ca.pem file generated during certificate generation in order to verify the RADIUS server's identity. In case the identity presented by the RADIUS server changes at any point, the supplicant fails to connect, and re-presents the user with the prompt for network credentials. It is possible to connect without installing the ca.pem file, but one needs to specify "No CA certificate required" or "Do not validate". In this case the supplicant will send credentials to any RADIUS server for that SSID without verifying its identity. It is possible to avoid sending the real user name in the unencrypted outer tunnel, by specifying a different value (normally 'anonymous') in the "Anonymous identity" field.

On Mac OS, iPhone and Windows 10 supplicants, when connecting to the SSID for the first time, the server certificate's details are presented to the user and the user is asked if they want to trust the server. In case the identity presented by the RADIUS server changes at any point, the user will be prompted with a message, not containing any reasonable warning, sadly, that looks identical to the one displayed when connecting for the first time, where a user is extremely likely to press Trust/Connect once again (on Windows 10, the message also advises the user to connect if they are in a location where said SSID is expected to be present). On Windows 10, in case the user does press Connect again, the supplicant stores both identities and thereon connects without further warning to any server presenting any of those identities, on Mac OS and iPhone this has not been tested. On Mac OS and Windows 10, it is also possible to copy the ca.pem file and install it like with the Linux supplicant, on iPhone, this did not seem to have any effect. On Windows 10, this does not change the behavior in case the server's identity changes, on Mac OS this has not been tested, presumably the same. It seems to be possible, but greatly complicated (involving installing a software from the App Store, and using it to create a configuration profile which then needs to be saved to a file, copied and imported onto the supplicant device) on Mac OS and iPhone to configure the supplicant to not send the real user name in the unencrypted outer tunnel. On Windows 10 this is somewhat easier.