How To setup govroam for government organisations Cookbook

Setting up govroam is very similar to eduroam. The eduroam wiki itself may be used to configure for govroam. In general if an instruction states 'eduroam' for the govroam setup this must be 'govroam'.

Original source and credits:
https://wiki.terena.org/display/H2eduroam/How+to+deploy+eduroam+on-site+or+on+campus#Howtodeployeduroamon-siteoroncampus-IdPandSPRADIUSinfratructure

CookBook version: 1.2
Date: May 22th 2020

Table of Contents

1 govroam in a nutshell

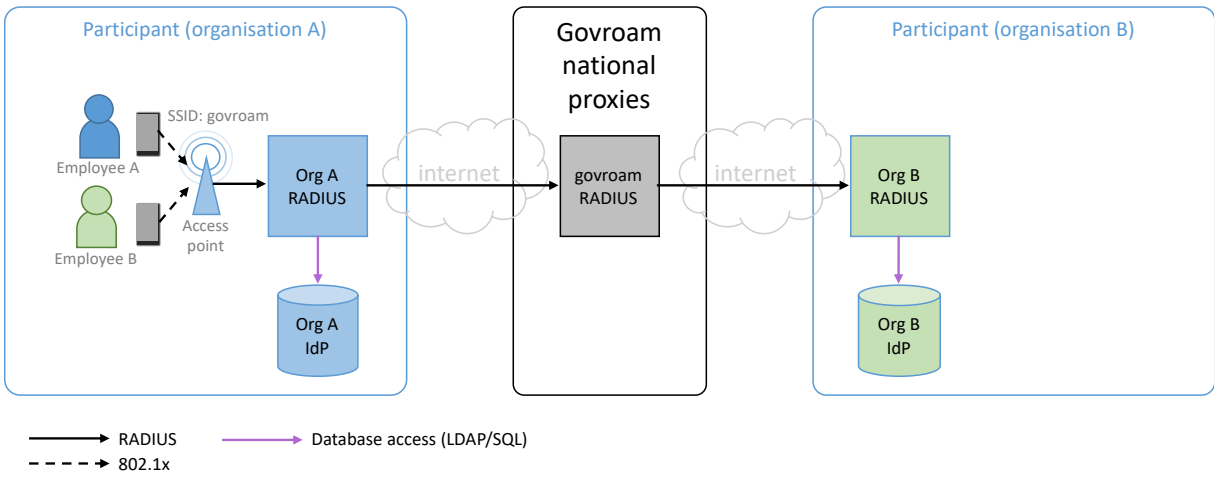
1.1 General overview

govroam stands for governmental roaming. It offers users from participating govroam organisation secure Internet access at any other government participating location. The govroam architecture that makes this possible is similar to the eduroam (educational roaming) worldwide roll-out and over 15 years over experience. govroam as well as eduroam are based on a number of technologies and agreements, which together provide the govroam user experience: "open your laptop and be online".

The crucial agreement underpinning the foundation of govroam involves the mechanism by which authentication and authorisation works:

- The authentication of a user is carried out at their Identity Provider (IdP), using their specific authentication method.
- The authorisation decision allowing access to the network resources upon proper authentication is done by the Service Provider (SP), typically a Wi-Fi network or Wi-Fi hotspot.

In order to transport the authentication request of a user from the Service Provider to his Identity Provider and the authentication response back, a world-wide system of RADIUS servers is created. Typically, every Identity Provider deploys a RADIUS server, which is connected to a local user database. This RADIUS server is connected to a federation level RADIUS server, which is either in turn connected to the upstream RADIUS server infrastructure or can connect to other RADIUS servers dynamically (using the protocol RADIUS/TLS). Because users are using usernames of the format "user@realm", where realm is the IdP's DNS domain name often of the form institution.tld (tld=top-level domain; both country-code TLDs and generic TLDs are supported), the RADIUS servers can use this information to route the request to the appropriate next RADIUS server until the IdP is reached. An example of the RADIUS hierarchy is shown in Figure 2.1, where Employee B is a guest in the network of Service Provider organisation A, which requests a check of the identity of Employee B over the govroam RADIUS infrastructure at the Identity Provider as configured within organisation B:



To transfer the user's authentication information securely across the RADIUS-infrastructure to their IdP, and to prevent other users from hijacking the connection after successful authentication, the

access points or switches deployed by the SP use the IEEE 802.1X standard that encompasses the use of the Extensible Authentication Protocol (EAP). EAP is a container that carries the actual authentication data inside, the so-called EAP methods. There are many EAP methods an IdP can choose from.

govroam requires that the chosen EAP method must allow:

- mutual authentication (i.e. the user can verify that he is connected to "his" IdP whereever the user is
- encryption of the credentials used (i.e. only the user and his IdP will see the actual credential exchange; it will be invisible to the Service Provider and all intermediate proxies)

Some popular EAP methods in use in govroam are:

- PEAP ("Protected EAP") - a Microsoft protocol that establishes a TLS tunnel, and sends usernames and passwords in MS-CHAPv2 hashes inside)
- TTLS ("Tunneled TLS") - an IETF protocol that establishes a TLS tunnel, and sends usernames and passwords in multiple configurable formats inside)
- TLS ("Transport Layer Security") - an IETF protocol that authenticates users and the IdP with two X.509 certificates
- FAST ("Flexible Authentication via Secure Tunneling") - a Cisco protocol that establishes a TLS tunnel, and sends usernames and passwords in a custom way inside)

RADIUS transports the user's name in an attribute User-Name, which is visible in clear text to all intermediate hosts on the way. Some EAP methods allow to put a different User-Name into the RADIUS packet than in the EAP payload. In that case, the following terms are used:

- outer identity: this is the User-Name in the RADIUS packet and visible to all intermediate parties. It is recommended to configure this to not use the actual user identity, but 'anonymous@realm.tld' instead
- inner identity: this is the actual user identification. It is only visible to the user himself and the Identity Provider

When using such EAP methods, and activating this option, the real username is not visible in RADIUS (it will only see the outer identity). Doing so will enhance the user's privacy, and is encouraged. Outer identities should be in the format "@realm" (nothing left of the @ sign, but the realm is the same as with the actual username). The realm part still must be the correct one as it is used to route the request to the respective Identity Provider. Once the IdP server decrypts the TLS tunnel in the EAP payload, it gets the inner identity and can authenticate the user.

The realm should end on a publicly available Top Level Domain (TLD). For the Netherlands, this is usually .nl.

After successful authentication by the Identity Provider and authorisation by the Service Provider, this SP grants network access to the user, possibly by placing the user in a specific VLAN intended for guests.

In the next chapter the various elements of this architecture and their functions is described.

1.2 Elements of the govroam infrastructure

1.2.1 Confederation top-level RADIUS Server (TLR)

The confederation top-level RADIUS Servers, at the time of writing, are located in the Netherlands and Denmark for the European confederation, and Australia and Hong Kong for the Asian and Pacific region. Each have a list of connected country domains (.nl, .dk, .au, .cn etc.) serving the appropriate National Roaming Operators (NROs). They accept requests for federation domains for which they are authoritative, and subsequently forward them to the associated RADIUS server for that federation (and transport the result of the authentication request back). Requests for federation domains they are not responsible for are forwarded to the proper confederation TLR.

1.2.2 Federation-Level RADIUS servers (FLRs)

A federation RADIUS server has a list of connected IdP and SP servers and the associated realms. It receives requests from the confederation servers and IdP/SP it is connected to and forwards them to the proper server, or in case of a request for a confederation destination to a confederation server.

1.2.3 IdP and SP RADIUS infrastructure

govroam IdPs operate a RADIUS server which is responsible for authenticating its own users, by checking the credentials against a local identity management system.

govroam SPs operate RADIUS capable equipment like Access Points or switches (see below). Large SPs typically also deploy an own RADIUS server, which is then responsible for forwarding requests from visiting users to the respective federation RADIUS server. Upon proper authentication of a user the SP RADIUS server may assign a VLAN to the user. Small SPs which do not require VLAN assignments can connect their RADIUS equipment directly to their FLR server, if the FLR permits that mode of operation.

Institutions which opt to be govroam IdP and govroam SP at the same time can have one RADIUS server that fulfils both roles simultaneously. This is the most popular deployment model in govroam.

Note that the IdP RADIUS server requires more configuration than the SP-part of the RADIUS server. Whereas the other RADIUS servers merely proxy requests, the IdP server also needs to handle the requests, and therefore needs to be able to terminate EAP requests and perform identity management system lookups.

1.2.4 Identity Management System

The Identity Management System of govroam IdPs contains the information of the end users; i.e. usernames and passwords. They must be kept up to date by the responsible IdP. An IdP RADIUS server will query the Identity Management system to perform the actual authentication for a user as he tries to log in.

1.2.5 Supplicants

A supplicant is a piece of software (often built into the Operating System but also available as a separate program) that uses the 802.1X protocol to send authentication request information using EAP. Supplicants are installed and operate on end-user computing devices (e.g. notebooks, PDAs, WiFi-enabled cell phones, and so on).

1.2.6 Access Points

Access Points are Wireless LAN access devices conformant to IEEE 802.11 (wireless) networking protocols and need to be IEEE 802.1X capable. Support for commonly available frequency bands, current ones and those standardised and licensed appropriately, is strongly advised. They must be able to forward access requests coming from a supplicant to the SP RADIUS server, to give network access upon proper authentication, and to possibly assign users to specific VLANs based on information received from the RADIUS server. Furthermore, Access Points exchange keying material (initialisation vectors, public and session keys, etc.) with client systems to prevent session hijacking.

Please note that sometimes the shortcut "802.11x" is used for the set of wireless IEEE protocols like 802.11b, 802.11ac and so on. This shortcut is not to be confused with the 802.1X protocol that focuses solely on access control.

1.2.7 Switches

Switches need to be able to forward access requests coming from a supplicant to the SP RADIUS server, to grant network access upon proper authentication and to possibly assign users to specific VLANs based on information received from the RADIUS server.

1.2.8 VPN

The RADIUS protocol combined with EAP provides a safe way to tunnel authentication requests. To add additional security, the RADIUS packets can be encrypted by radsec on the RADIUS level, or sent over a pre-setup VPN tunnel towards the national govroam servers (upon request during the onboarding)

1.2.9 Firewalls and other infrastructure

Although a firewall should be simply configured to allow for the RADIUS traffic to pass to the govroam servers and back, from experience, a couple of recommendations are given:

- The firewall should be configured to allow both incoming as well as outgoing connections (of course)
- The firewall should, if it is 'application aware', be checked to allow the detected 'application'
- The firewall should not block segments of fragmented packets

If NAT is used to connect the RADIUS server(s) to the internet, please be aware that the NAT-address should have a fixed mapping to the public IP-address. Also make sure that the mapping is symmetrical, so the same public IP address is used for both incoming as well as outgoing RADIUS traffic (the might, if not explicitly configured, default to a different public IP address).

2 govroam set up on government location: IdP and SP

The following sections provide detailed information for the two roles govroam IdP and govroam SP, respectively.

The govroam IdP section explains the administrative obligations for an govroam IdP, the setup of several popular RADIUS servers, and means to provision configuration details of supplicants to end users.

The govroam SP section explains general basics of wireless LAN deployment, the administrative obligations for an govroam SP, and the setup of several popular vendor Wi-Fi environments for use in govroam.

2.1 govroam SP

Basic deployment considerations for wireless LANs.

A govroam wireless network is a wireless network. This sounds trivial, but it is important to keep in mind that

- a poorly managed Wireless LAN won't magically become better by naming it govroam. Before diving into govroam -specific configuration, make sure you understand how to manage
    - Wi-Fi coverage
    - bandwidth requirements
    - enough DHCP addresses to accommodate all clients
- by naming the network govroam, you are becoming part of a recognised brand. Arriving users will think of this being an govroam network, with a set of expectations for such networks. If your wireless network fails to deliver in the points mentioned above, users will consider this an govroam failure and your installation will hurt the (global) brand govroam, not only your own site and users.

This section provides general advice regarding wireless LAN deployment. It is not meant as a replacement for further literature; there are many books and online publications regarding good wireless LAN planning, and you are encouraged to familiarise yourself with this topic.

Please note that the technologies that govroam authentication relies on (802.1X, RADIUS), stem from the world of wired connectivity, also summarised as 'Network Access Control' (NAC), and can be used for the purpose of providing safe access to wired networks as well.

2.1.1 Set up of Wi-Fi hotspots (Cisco)

All of the solutions presented below support the basic requirements for a govroam SP: support for IEEE 802.1X authentications, WPA2/AES or WPA3 support. When deploying govroam, deployers often want to make use of additional features such as multi-SSID support, dynamic VLAN assignment and others. Every section contains a table with a short overview of their support of such additional useful features.

Cisco (controller-based solutions)

Feature	supported?

Multi-SSID	yes
VLANs	yes
dynamic VLAN assignment	partial; not with Ipv6

The example configuration shown in this chapter was obtained from a Cisco 4400 Series Wireless LAN Controller.

2.1.2 Initial settings and defining the IP address

In the first phase the controller must be accessed through the Command Line Interface (CLI). When an IP address has been assigned to the controller, further configuration can be done using the web interface, but the CLI can be continued to be used.

Establish access to the controller by using a serial console and configure the initial settings for example as follows.

```
Welcome to the Cisco Wizard Configuration Tool
Use the '-' character to backup
System Name [Cisco_b2:e2:83]: <your_system_name>
Enter Administrative User Name (24 characters max): <your_username>
Enter Administrative Password (24 characters max): <your_password>
Re-enter Administrative Password                 : <your_password>

Service Interface IP Address Configuration [none][DHCP]: DHCP

Enable Link Aggregation (LAG) [yes][NO]: NO

Management Interface IP Address: esim. Xxx.yyy.zzz.1
Management Interface Netmask: <your_network_mask>
Management Interface Default Router: <your_router's_IP_address>
Management Interface VLAN Identifier (0 = untagged): <0 or 1>
Management Interface Port Num [1 to 2]: 1
Management Interface DHCP Server IP Address: esim. Xxx.yyy.zzz.2

AP Transport Mode [layer2][LAYER3]: <layer2 if controller and access points are on
same subnet; layer3 if routing in between>
AP Manager Interface IP Address: esim. Xxx.yyy.zzz.3

AP-Manager is on Management subnet, using same values
AP Manager Interface DHCP Server (xxx.yyy.zzz.2):

Virtual Gateway IP Address: xxx.yyy.zzz.www

Mobility/RF Group Name: <choose a suitable name if you have more than one
controller. Otherwise, don't care>

Enable Symmetric Mobility Tunneling [yes][NO]: NO

Network Name (SSID): <Define a test SSID at first>
Allow Static IP Addresses [YES][no]: no

Configure a RADIUS Server now? [YES][no]: no #Will be done later
Warning! The default WLAN security policy requires a RADIUS server.
Please see documentation for more details.

Enter Country Code list (enter 'help' for a list of countries) [US]: <your country
abbreviation>

Enable 802.11b Network [YES][no]: no
Enable 802.11a Network [YES][no]: YES
Enable Auto-RF [YES][no]: YES

Configure a NTP server now? [YES][no]: no #Will be done later
Configure the system time now? [YES][no]: no #Will be done later

Warning! No AP will come up unless the time is set.
```

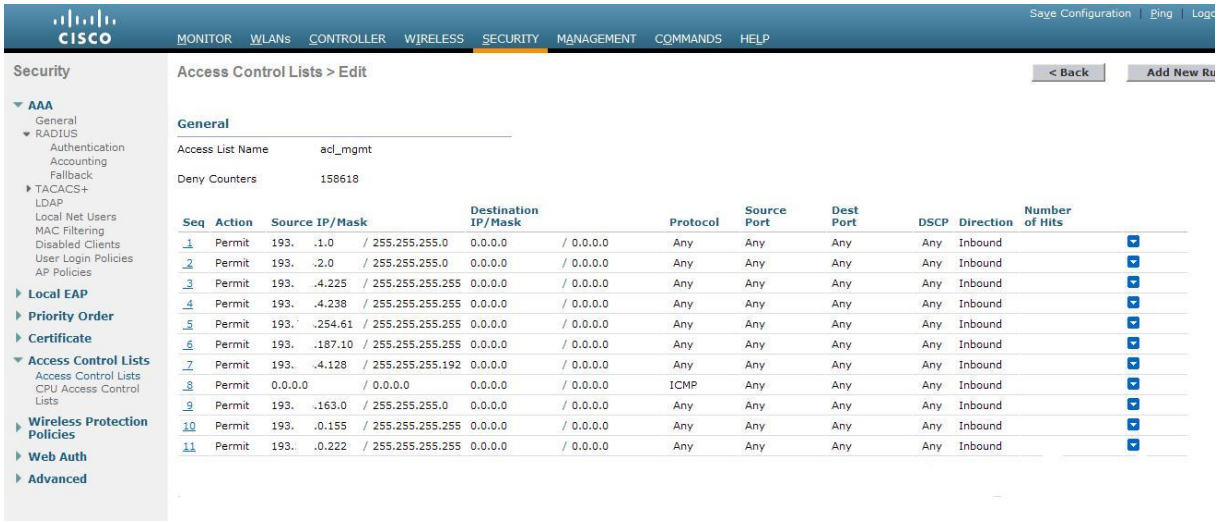
```
Please see documentation for more details.

Configuration correct? If yes, system will save it and reset. [yes][NO]: yes

#When the system has rebooted, familiarize yourself with the CLI by defining
(Cisco Controller) >config time ntp server 1 xyz.zyx.zzy.wyz
(Cisco Controller) >config time ntp server 2 xyz.zzz.zzy.wyz
```

2.1.3 Access Control Lists

After the initial setup, the access control (ACL) list needs to be configured, in order to prohibit unauthorized access to the controller. Choose SECURITY and then Access Control Lists | Access Control Lists and create an ACL by pressing New... The ACL shoud include at least

- the networks from which maintenance is carried out
- the address(es) of the monitoring server(s)
- the network(s) from which the Aps and the WLAN clients get their addresses
- the address(es) of the RADIUS server(s)
- a rule to always answer ping commands

An example of an ACL is shown below. Inbound means packets towards the controller and outbound means packets towards the WLAN
clients.



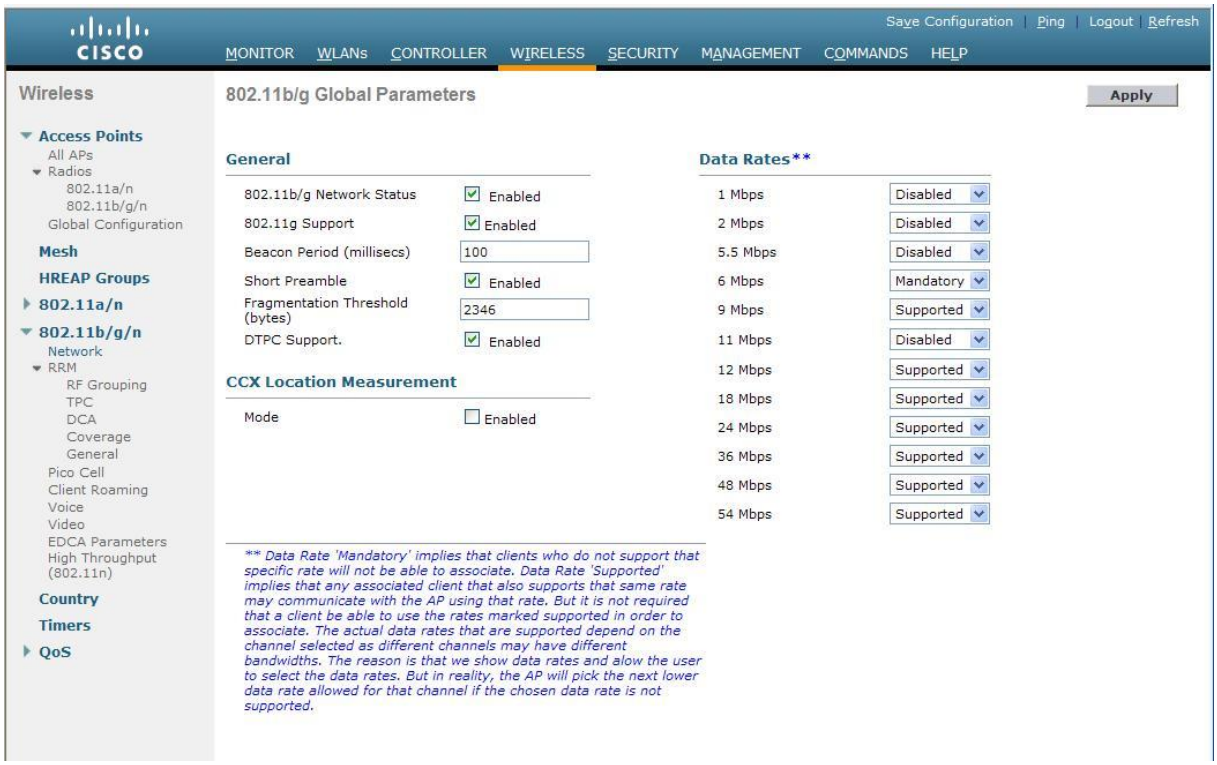
After you have specified the ACL you need to take it into use by first selecting Access Control Lists from the side bar and by choosing your ACL and specifying the CPU ACL Mode to *Wired* or *Both*.

2.1.4 Access Point configuration: Cisco

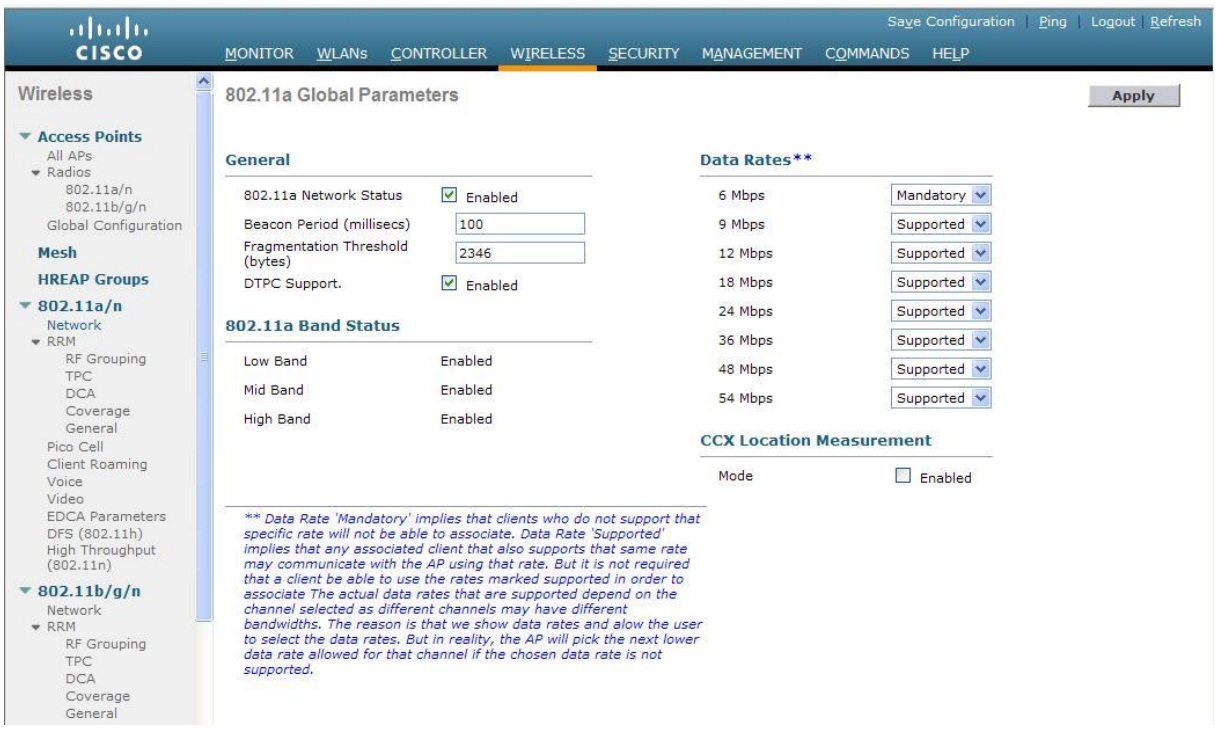
If the access points are connected to the same subnet as the controller, they will automatically find the controller and connect to it. If this is not the case, the IP address of the controller must be find from the name server by the name CISCO-LWAPP-CONTROLLER. Once the access point has found the controller, it stores the IP of the controller, and it can connect to it from any network, as long as the network allowed access in the ACL (see previous section).

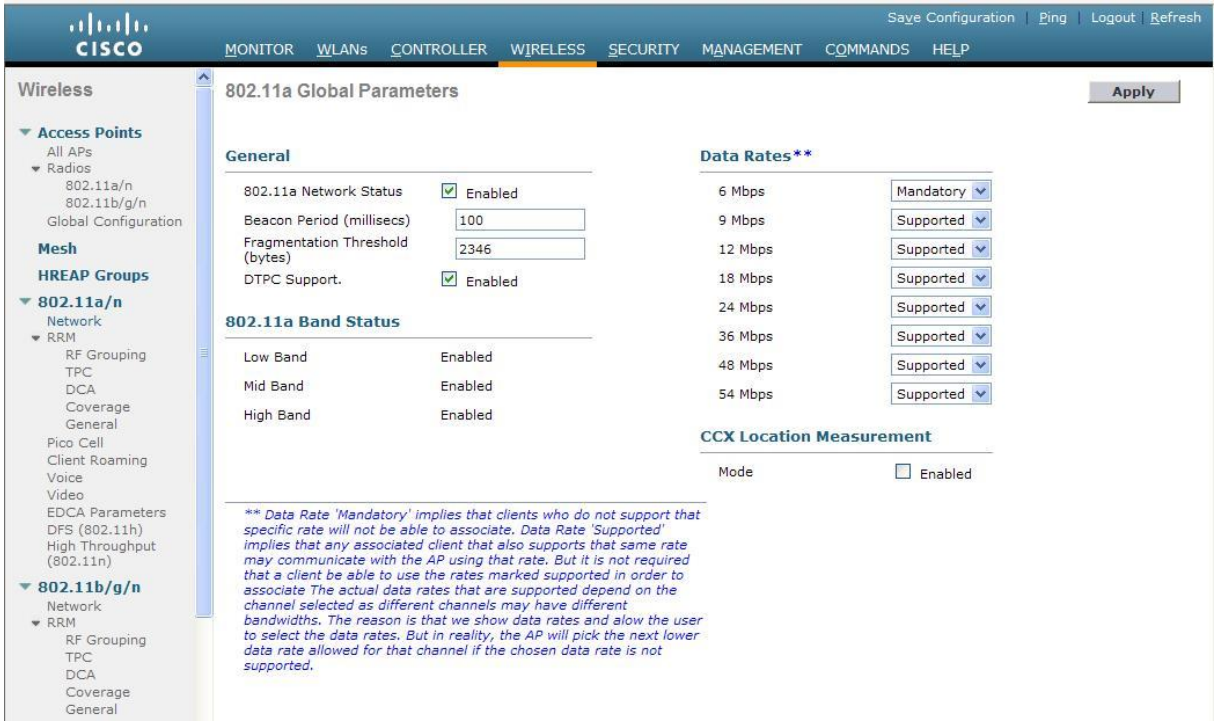
The next step is to define the wireless network, which has to be done separately for 2,4 GHz and 5 GHz. First, choose WIRELESS and then 802.11b/g/n | Network. Enabling the 802.11b-standard will result in less available capacity on your network and therefore it is recommended to enable only the standards 802.11g and 802.11n. Enable 802.11g according to the figure shown below. If you want to support also the 802.11-b standard, set _Mandatory_ for the lowest 802.11b-rate that you want to

support (1 Mbps, 2 Mbps, 5.5 Mbps or 11 Mbps), set _Supported_ for all data rates higher than this rate and _Disabled_ for all rates lower than this rate. If 802.11b needs to be supported, it may pay off to disable the lowest rates, in order to avoid clients being attach to an AP far away, unwilling to roam.

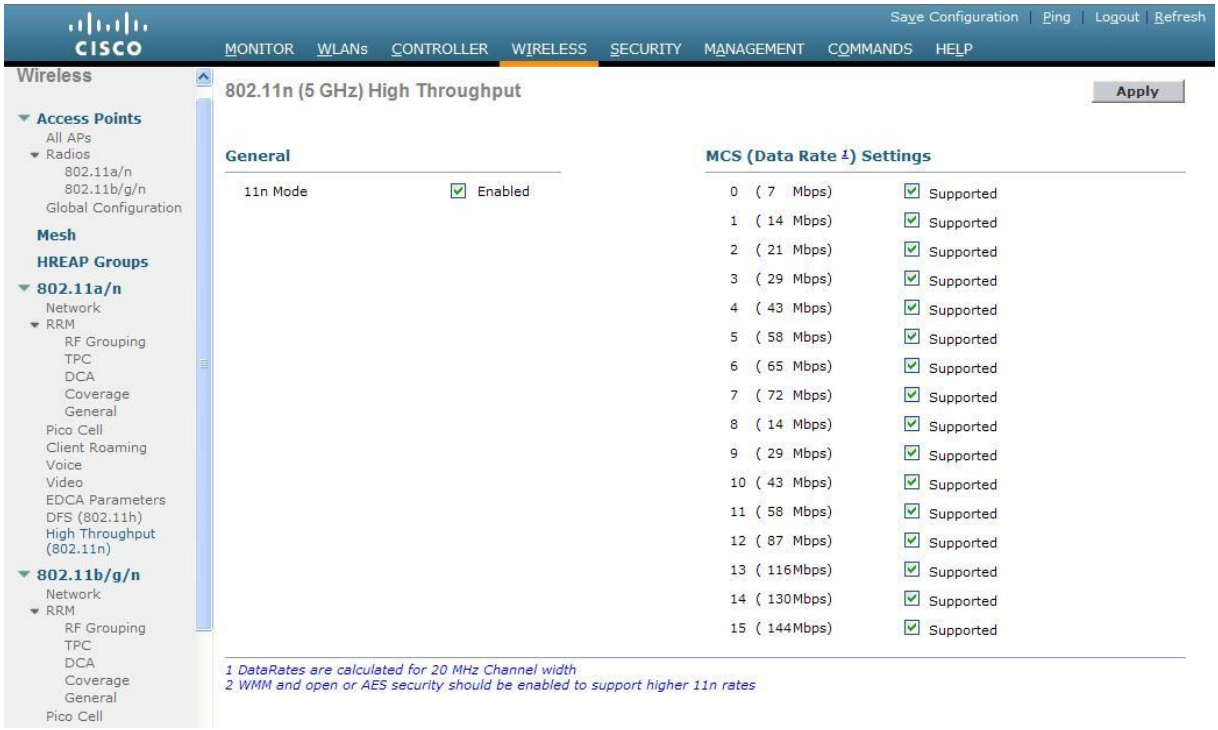


Next, switch to enable the standard 802.11a for 5 GHz by selecting 802.11a/n | Network. Configure the settings according to the figure below.





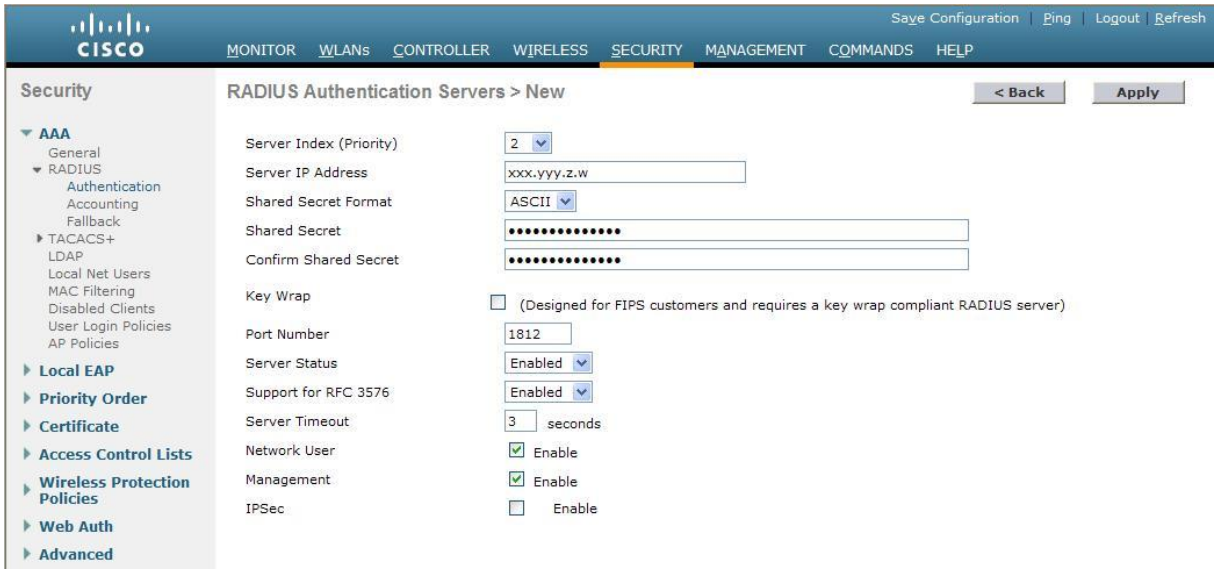
The only standard left to enable is the standard 802.11n. You can choose to enable it for either 2,4 GHz or 5 GHz. It has been suggested that 802.11n is enabled only on the 5 GHz band, in order to utilise the radio resources effectively, see for example the (eduroam) Campus Best Practice document on “WLAN network planning and setup” Chapter 6.3.http://www.terena.org/activities/campus-bp/pdf/gn3-na3-t4-wlan-network-planning.pdf. However, newer 802.11 standards like 802.11ax (Wifi6) improve bandwidth and performance in the 2.4GHz bands significantly, continuing the usefulness of the 2.4GHz band. To enable 802.11n in the network select 802.11a/n | High throughput (802.11n) and/or 802.11b/g/n | High throughput (802.11n) and configure the settings according to the figure below.



At this point you have enabled the radios, but you have not yet defined any network, so don't try to use the access points just yet.

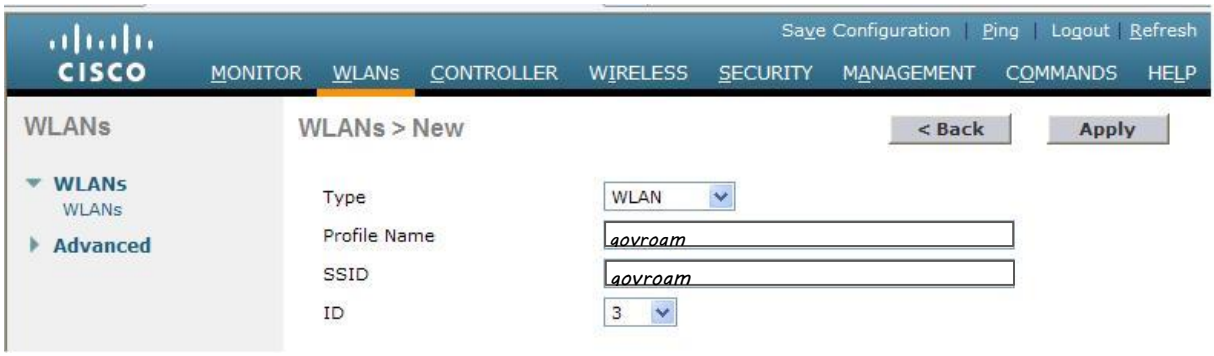
2.1.5 Defining the RADIUS server

Define the RADIUS server to be used in the govroam network by selecting SECURITY and then AAA | RADIUS | Authentication. Define the IP address, the shared secret and the other parameters according to figure. Please note that your first server will naturally have a server index of one.



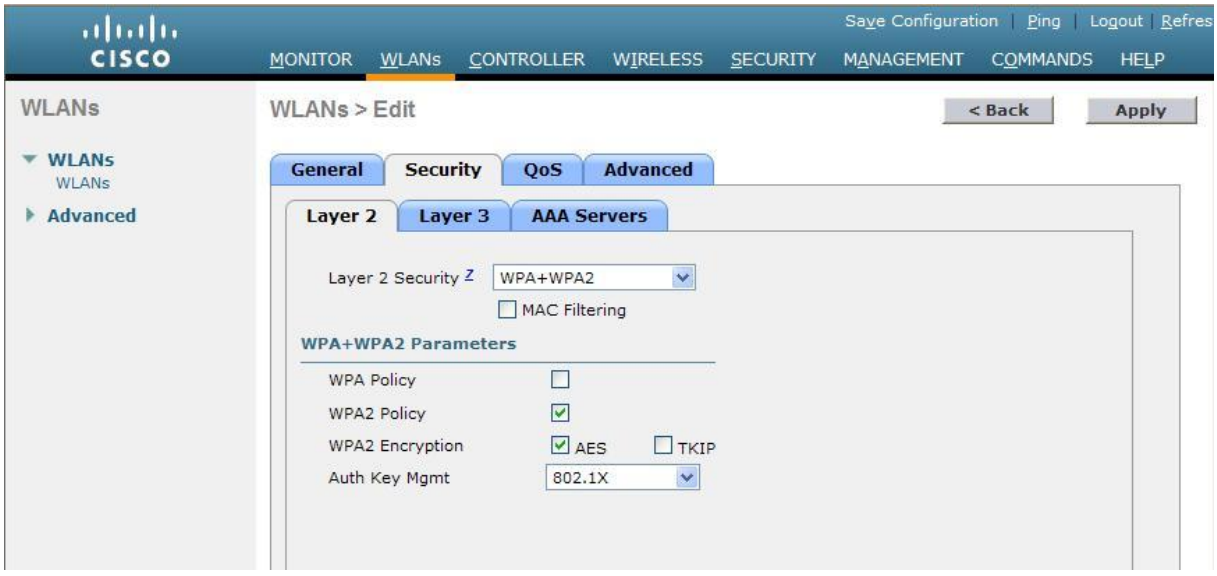
2.1.6 Defining a wireless network

Select WLANs and then WLANs | WLANs from the sidebar. Create a new network and name it as shown in the figure below.

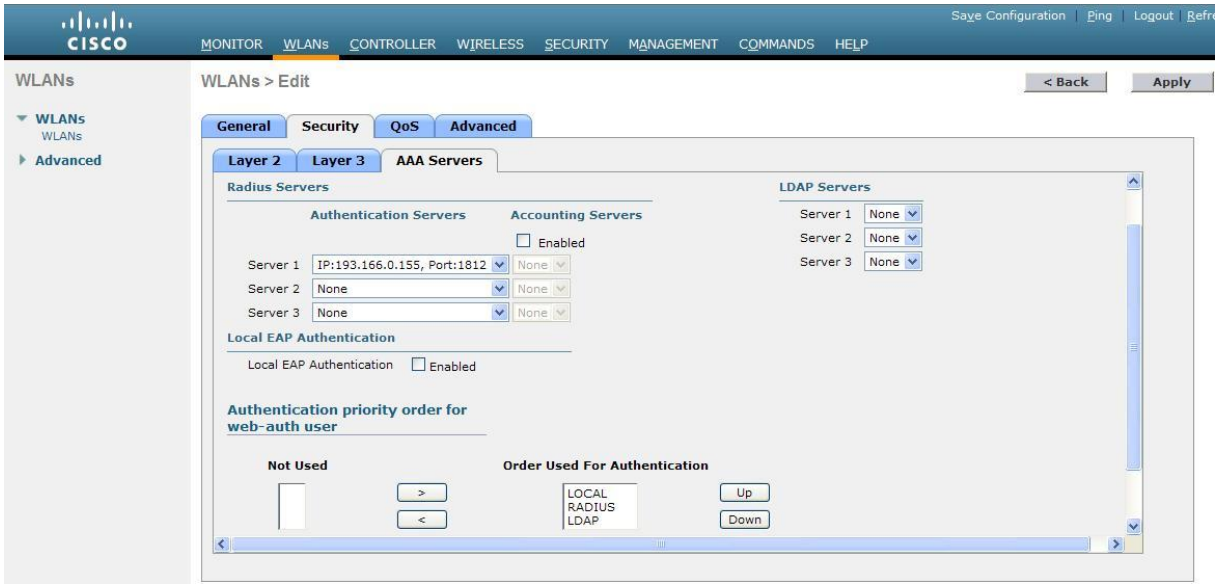


After defining the govroam network, click on the WLAN ID number to start defining the settings for the network. Set the General settings according to the figure below, then click the Security tab.

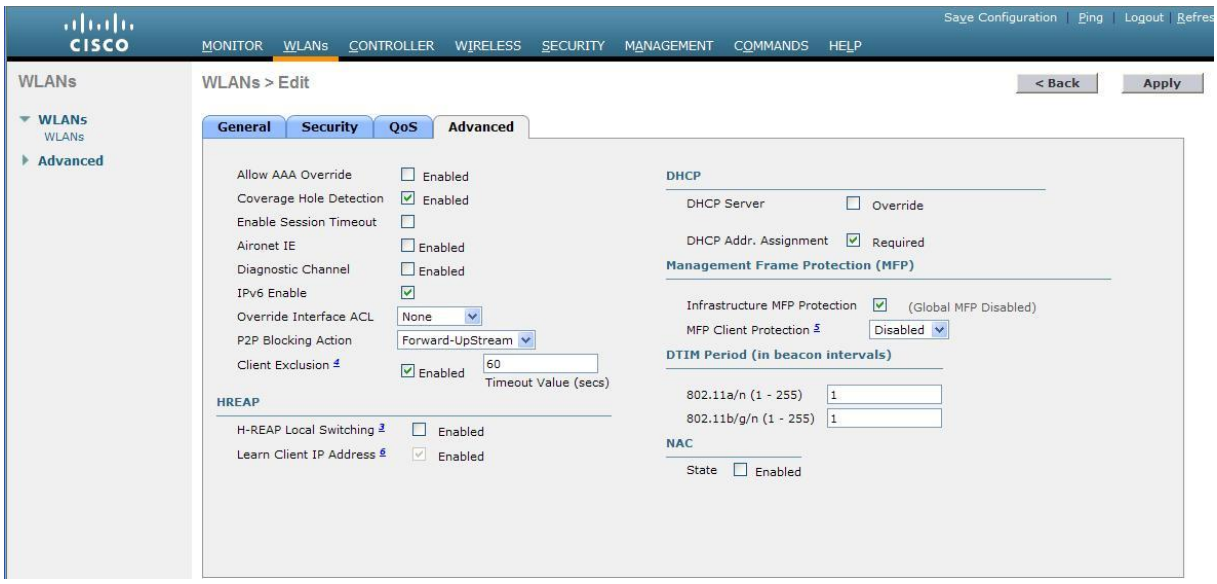
In order to enable only WPA2-AES, fill in the security settings as shown in the figure below.



After this, click on the AAA Servers tab and select the RADIUS server that you defined earlier to be used in govroam.



Next, click on the QoS tab and make sure that you have set the WMM Policy to either Required or Allowed. Otherwise, the higher transmission rates associated with the 802.11n-standard will not work. Then select the Advanced tab and adjust the settings as shown in the figure below. By choosing the parameter P2P Blocking Action to have the value Forward-UpStream, you can prevent WLAN clients to communicate directly, without involving the AP, as recommended in the Campus Best Practice document on “WLAN Information Security” Chapter 2.2 and 2.3. MFP Client Protection is known to have caused problems and can be disabled.



At this stage, click Apply. In the Advanced-tab, the Client Exclusion timeout value was set to 60s. While this is a suitable value, the rules for client exclusion are a bit too strict. Hence, it pays off to adjust the rules by selecting SECURITY and then Wireless Protection Policies | Client Exclusion Policies from the sidebar and uncheck all other options except for “IP Theft or IP Reuse”.

These are the basic settings for the Cisco controller. More advanced settings can be found from the upcoming Campus Best Practice document on “WLAN infrastructure”, to be published in the first half of 2011.

2.1.7 Cisco (stand-alone Aps with IOS)

Feature	supported?
Multi-SSID	yes
VLANs	yes
dynamic VLAN assignment	yes

The following equipment was used for this section:

Cisco AP 1200 Series (802.11g Radio).

The configuration examples used in this document were tested and made on a Cisco Series 1200 with an 802.11g Radio Module and with the following Cisco software:

```
IOS Version:
Cisco IOS Software, C1200 Software (C1200-K9W7-M), Version 12.3(8)JA2,
RELEASE SOFTWARE (fc1)

Bootloader:
C1200 Boot Loader (C1200-BOOT-M) Version 12.2(8)JA, EARLY DEPLOYMENT RELEASE
SOFTWARE (fc1)
```

2.1.8 Setting the Name and IP address

First, an IP address on the BVI interface (the IP address that this Access Point will have for accessing resources like the RADIUS server) needs to be configured. Also a unique name for this Access Point (ap1200) will be configured.

```
Ap#configure terminal
ap1200(config)#hostname ap1200
ap1200(config)#interface BVI 1
ap1200(config-if)# ip address 192.168.10.200 255.255.255.0
```

2.1.9 RADIUS/AAA section

In the authentication, authorisation and accounting configuration parameters (AAA), at least one group needs to be defined (radsrv), which will be assigned later for the several AAA operations. More groups can be defined if needed for various purposes; one for authentication, another for accounting, and so on. In this example the
RADIUS server has the IP address 192.168.10.253.

```
ap1200(config)#aaa new-model
ap1200(config)#radius-server host 192.168.10.253 auth-port 1812 acct-port 1813 key
<secret>
ap1200(config)#aaa group server radius radsrv
ap1200(config-sg-radius)#server 192.168.10.253 auth-port 1812 acct-port 1813
ap1200(config-sg-radius)#!
Ap1200(config-sg-radius)#aaa authentication login eap_methods group radsrv
ap1200(config)#aaa authorization network default group radsrv
ap1200(config)#aaa accounting send stop-record authentication failure
ap1200(config)#aaa accounting session-duration ntp-adjusted
ap1200(config)#aaa accounting update newinfo periodic 15
ap1200(config)#aaa accounting network default start-stop group radsrv
ap1200(config)#aaa accounting network acct_methods start-stop group radsrv
```

2.1.10 Configuring the SSIDs

For each SSID one dot11 ssid <SSID NAME> must be configured. In this section the default VLAN for the SSID will be configured as well as the authentication framework, the accounting and, if desired, the SSID to be broadcast (guest-mode).

```
Ap1200(config)#dot11 ssid eduroam
ap1200(config-ssid)#vlan 909
ap1200(config-ssid)#authentication open eap eap_methods
ap1200(config-ssid)#authentication network-eap eap_methods
ap1200(config-ssid)#authentication key-management wpa optional
ap1200(config-ssid)#accounting acct_methods
ap1200(config-ssid)#guest-mode
```

More SSIDs can be configured. An open SSID for giving information about the institution and/or how to connect to the govroam SSID:

```
ap1200(config)#dot11 ssid guest
ap1200(config-ssid)#vlan 903
ap1200(config-ssid)#authentication open
ap1200(config-ssid)#accounting acct_methods
```

2.1.11 The Radio Interface

Now the configured SSID's will be mapped to the radio interface, and it will be specified what ciphers will be used/allowed on each VLAN. If dynamic VLANs are planned, the ciphers for those VLANs must also be configured even if there is no direct mapping on any SSID (this example shows the usage of the VLANs 906 and 909 for eduroam users)

```
ap1200(config)#interface Dot11Radio 0
ap1200(config-if)# encryption vlan 906 mode ciphers aes-ccm tkip wep128
ap1200(config-if)# encryption vlan 909 mode ciphers aes-ccm tkip wep128
ap1200(config-if)#ssid eduroam
```

To bind extra SSID's the previous command, for each SSID to be bound, needs to be repeated.
The following command sets the maximum time (e.g. 300 seconds, which is recommended) for rekeying/reauthentication:

```
dot1x reauth-period 300
```

2.1.12 VLAN interfaces

For each VLAN to be used for wireless clients, two virtual interfaces need to be defined: one on "the air" (DotRadio) and another on the "wire" (FastEthernet) then they need to be bridged together with the same bridge group. These VLANs are always tagged with the proper VLAN identifier.

An administrative VLAN needs to be configured as well (for maintenance/management and authentication/accounting traffic). This VLAN is usually untagged (the command defining the VLAN has to be suffixed with the keyword "native") and belongs to bridge-group 1. The Radio virtual interface for this VLAN does not need to be defined since the default will keep the physical interface (Dot Radio 0) in bridge-group 1.

Because VLANs can be from 1 to 4094 and bridge-groups from 1 to 255, it is not necessary to have the same bridge-group id as the vlan id.

```
Ap1200(config)#interface dot11Radio 0.903
ap1200(config-subif)#encapsulation dot1Q 903
ap1200(config-subif)#bridge-group 3
ap1200(config)#interface fastEthernet 0.903
```

```
ap1200(config-subif)#encapsulation dot1Q 903
ap1200(config-subif)#bridge-group 3
ap1200(config)#interface dot11Radio 0.909
ap1200(config-subif)#encapsulation dot1Q 909
ap1200(config-subif)#bridge-group 9
ap1200(config)#interface fastEthernet 0.909
ap1200(config-subif)#encapsulation dot1Q 909
ap1200(config-subif)#bridge-group 9
```

2.1.13 The multiple (dynamic) VLAN assignment

The example configuration above did not configure dynamic VLAN assignment. Availability of this feature varies between models of the 1200 Series, so please exercise caution when procuring if you wish to make use of this feature. If multiple VLANs are configured on the Cisco AP, it is mandatory to associate each SSID to at least one VLAN, otherwise the Access Point will not activate the SSID's. It is possible however, to put different users who are connected to the same SSID (e.g. eduroam) on different VLANs based, for instance, on the user profile. To activate this feature it is necessary to enter

```
"aaa authorisation network default group radiusgroup"
```

in the Access Point's configuration. The AP then gives priority to the VLANs returned by RADIUS over the ones statically associated with the SSID. This enables the feature dynamic VLAN assignment.

Cisco's Access Points require that two virtual interfaces (a radio and an Ethernet port interface) are configured for each VLAN. If, for example, four VLANs are to be used for eduroam users (for students, admin staff, teachers and visiting eduroam users from other institutions for example) then it is necessary to define one Dot11Radio0.vlanID, and one FastEthernet0.vlanID, and ensure that both have the same encapsulation dot1Q vlanID and the same bridge-group for each VLAN.

Two commands that are also needed are the below, otherwise the access point will not change the user to the received VLAN:

```
encryption vlan vlanID mode ciphers aes-ccm

broadcast-key vlan vlanID change 600 membership-termination capability-change
```

2.2 govroam IdP

2.2.1 Selecting EAP types

choices

The decision which EAP type(s) to deploy on your govroam IdP depends on several factors:

- Capabilities of your Identity management backend
- Types of devices you want to support

2.2.2 Choices depending on the Identity Management System

Regarding the identity management backend, the most fundamental differentiation between EAP types is the type of credential they support.

- Does your identity management backend support X.509 Client Certificates? Then you can use EAP-TLS.
- Does your identity management backend use username/password combinations?
- Does it store the passwords as either clear text – or – encrypted as NT-Hash? Then you can use EAP-TTLS, PEAP, EAP-FAST, EAP-PWD and more.
- Does it store the passwords in a different crypt format? Then you can use EAP-TTLS only.

As you see, the decision is largely dependent on your identity management system; so your choices may be limited. As a more concrete advice for some IdM backends:

- Microsoft Active Directory: stores passwords as NT-Hashes.

In all cases, the Top-Level Domain of the identities' realm should be a publicly available (DNS) TLD. For Dutch organisations this is typically ".nl". The national govroam RADIUS servers need this for proper dispatching of authentication requests. This means that non-compliant TLD's like .local or .intra cannot be used.

The use of ".local" as top level suffix usually stems from a Microsoft Windows Active Directory based environment. Changing user identities is out of scope of the set-up of a govroam RADIUS peering. However, from experience, a set of recommendations are provided for your convenience.

The use of ".local" has been deprecated since Windows Server 2003. It is also not allowed in X.509 certificates that public Certificate Authorities can provide. If your 19organisation19 still uses ".local" as a top level suffix, you have a couple of options:

- *Comply to recommendations and best practices for Microsoft environments and provide your users with a UPN ending on a publicly available top-level domain name (e.g. ".nl"). Moving forward can have impact on many aspects of the IT landscape but ensures that user identity management is based on industry standards.*
- *Translate the domain name (and optionally username) in your RADIUS server. This feature is not available in Microsoft NPS*
- *Introduce a separate identity in the form of a URI ending on a publicly available top-level domain name (e.g. ".nl")*

2.2.3 Anonymous outer identities

Almost all EAP types support the use of anonymous outer identities. The primary use of anonymous outer identities is for better preservation of privacy for your users; a properly configured supplicant will then not even reveal the real username of the user to the visited govroam SP; instead, the username is replaced with a dummy value.

This feature needs protocol support by the EAP type in question; the basic idea is that there have to be two stages of communicating the client identity:

- one identity, the outer identity, is used to route the user's login request from the govroam SP via the govroam RADIUS path to the govroam IdP
- the second, "inner" identity, is only revealed inside a cryptographically protected tunnel to the IdP

Since the outer identity is only needed for routing purposes towards the IdP, the local username part does not have to be accurate and can be obfuscated. The IETF-suggested way of obfuscating the username is to leave it empty; but it can just as well be replaced with "anonymous", "anon" or similar. However, the realm part (i.e. behind the @ sign) always needs to be accurate because it contains the routing information.

The inner identity always needs to be fully accurate, because it is used to authenticate the user. It does not necessarily have to contain an @ sign at all, because that username is local to the IdP and is only seen and evaluated there.

Example:

- Outer identity: anonymous@gemeente.nl
- Inner identity: stefan.winter

For govroam request routing, the part @gemeente.nl of the outer identity is used to route the request to the gemeente.nl realm and to establish a secure tunnel; while the real username inside this tunnel which is looked up in a user database is "20unnel.winter".

Here is a break-down of anonymous outer identity support for some popular EAP types:

EAP-Type	Support for anonymous outer identities
EAP-TTLS	yes
PEAP	yes
EAP-FAST	yes
EAP-TLS	support in protocol, but not typically available in supplicants
EAP-PWD	no

If the EAP type allows for the use of outer identities, it is a client device configuration option to either make use of them or not; there is little you as an IdP can do to force the use of anonymous outer identities (except for providing and encouraging the use of pre-configured installers which will then make all the necessary settings on the client device automatically).

2.2.4 Choices depending on the envisaged devices

The landscape of wireless-enabled devices is rather heterogeneous, and support for EAP types varies. Ideally, you should survey which types of devices you should come to expect among your user base,

check the capabilities of these devices, and make an informed decision regarding the EAP type of choice.

However, the EAP protocol is flexible enough to handle multiple EAP types: if your IdM backend can support the use of multiple EAP types, then you can configure all the supported EAP types. In that case, you have to select a "default" EAP type – it should be set to the EAP type with the broadest support in your client base.

Now, assuming you have the option of configuring a range of EAP types *and* your clients support that same range, which of these types should you prefer?

- We suggest the use of PEAP over EAP-TTLS for it does a mild amount of protection of the user password inside the secure tunnel.
- If you cannot support PEAP, consider to allow TTLS-PAP **and** the more unusual variant TTLS-GTC (initially Generic Token Card; also used for passwords which are not savable on the client device). Some older devices (certain Symbian OS builds) support TTLS, but not PAP inside. Enabling TTLS-GTC will allow these devices to connect.

2.2.5 EAP Server certificate considerations

Almost all EAP types in govroam (with the exception of EAP-PWD) require an X.509 server certificate with which the RADIUS server identifies itself to the end user before the user sends his credentials to the server.

2.2.6 Consideration 1: Procuring vs. creating your own server certificate

In a generic web server context, server certificates are usually required to be procured by a commercial Certification Authority (CA) operator; self-made certificates trigger an "Untrusted Certificate" warning. It makes sense for browsers to have a pre-configured trust store with many well-known Cas because the user may browse to any website; and the operator of that website may have chosen any of those well-known Cas for his website. In an abstract notion, one can say: it is required to have many Cas in the list because the user device does not have all required information for certificate validation contained in its own setup; it misses the information "which CA did the server I am browsing to use to certify the genuinely one of his website?".

These considerations are not at all true in an EAP authentication context, such as a govroam login. Here, the end user device is pre-provisioned with the entire set of information it needs to verify this specific TLS connection: the IdP has a certificate from exactly one CA, and needs to communicate both that CA and the name of his authentication server to the end user. A trust store list from the web browser is thus insignificant in this context; certificates from a commercial CA are as valid for EAP authentications as are self-made certificates or certificates from a small, special-purpose CA. For a commercial CA, the installation of the actual CA file may be superfluous in some client operating systems (particularly those who make their "web browser" trust store also accessible for EAP purposes), but marking that particular CA as trusted for this specific EAP authentication setup still needs to be done.

Note that also root CA certificates have an expiry date. Both for commercial and private Cas please be aware that an exchange of the root CA certificate will require re-configuration of all your end-users' devices to accept the new CA. As a consequence: for commercial Cas, check their root CA's expiry date so you can make an informed decision whether you want to buy the certificate from

them or not. For your own private-use CA: choose a very long expiry date for the CA. Especially for commercial Cas, keep in mind that if you ever want to switch to a *different* CA as a trust anchor, all your end-user devices again need to be re-configured for that new root.

Configuration tools like govroam CAT enable to provision the chosen CA(s) and the expected server name(s) into client devices without user interaction. In that light, it does not make much difference whether to procure a server certificate from a commercial CA or to make your own; either way, configuration steps are necessary on the end-user device to enable and secure your chosen setup. With the conceptual differences being small, a number of secondary factors come into play when making the decision where to get a server certificate from:

- Do you have the necessary expertise to create a self-signed certificate; or to set up a private Certification Authority and issue a server certificate with it? Consider in particular the next "Consideration 2" which imposes some properties onto the certificates you need.
- Does your govroam NRO operate a special-purpose CA for govroam purposes, so that you could get a professionally crafted certificate without much hassle?
- Do your end-user devices all verify the exact server identity (issuing CA certificate AND expected server name)?

The third question is particularly important these days because some popular operating systems, particularly Android ones, do not allow to verify the expected server name. For such operating systems, using a commercial CA for the server certificate opens up a loophole for fraud: anyone with a valid certificate from this CA, regardless of the name in the certificate, can pretend to be the govroam authentication server for your end-user; which ultimately means the end-user device will send the user's login credentials to that unauthorised third-party. If you use a self-signed certificate or private CA however, which issues only one/very few certificates, and over which you have full control, then no unauthorised third party will be able to get a certificate in the first place, and thus can't fraud your users.

Another factor to consider when making the decision private vs. commercial CA is that of size and length of the EAP conversation during every login: with a private CA, you will be able to construct a certificate chain without intermediary CA certificates; requiring less bytes to be transmitted inside the EAP conversation (see Consideration 3, below). This results in fewer EAP round-trips and thus a faster authentication.

So, as a general recommendation: if you have the required expertise, it is suggested to set up a private CA exclusively for your IdP's govroam service. This CA should have a very long lifetime to prevent certificate rollover problems. The CA should issue only server certificates for your govroam IdP server(s). If you do not have that expertise, you should make use of your NROs special-purpose CA if it exists. If none of these work for you, a certificate from a commercial CA is the third option.

2.2.7 Consideration 2: Recommended certificate properties

Various end-user device operating systems impose different requirements on the contents of the server certificate that is being presented. Luckily, these requirements are not mutually exclusive. When creating or procuring a server certificate, you should check with the CA that its certificates satisfy as many of these requirements as possible to ensure broad compatibility with your users' devices. The list below does not include "standard" sanity checks applied to certificates; e.g. well-

formedness of the data, validity timestamps etc. These checks are done “as per usual” in every TLS connection.

The most important property of the server certificate is the name of the server. Since this certificate is not for a webserver, there is no necessity to put an actual hostname into the server name. Also, when an Identity Provider uses multiple servers for resilience reasons, then all these servers can and should have a certificate with the same name; and it may well be the identical certificate. Having different names for different servers means that end-user devices must be configured to trust multiple servers, which is more cumbersome than just having to configure one name string.

Some end-user device operating systems might (incorrectly) require the name to be parseable as a hostname; so it is a good idea to use a server name which parses as a fully-qualified domain name – the corresponding record does not have to exist in DNS though. The server name should then be both in certificate’s Subject field (Common Namecomponent) and be a subjectAltName:DNS as well.

The following additional certificate properties are non-standard and are of particular interest in the govroam context:

Property	Content	Remarks
server name	parses as fully-qualified domain name	Server certificates with spaces, e.g. “RADIUS Service of Foo University” are known to be problematic with some supplicants, one example being **Apple iOS 6.x**.
server name	Subject/CN == SubjectAltName:DNS	Some supplicants only consult the CN when checking the name of an incoming server certificate (**Windows 8 with PEAP**); some check either of the two; some new EAP types such as **TEAP** will only check SubjectAltName:DNS. Keeping the desired name in both fields in sync is a safe bet for futureproofness.
Server name	not a wildcard name (e.g “*.someidp.tld”)	Some supplicants exhibit undefined/buggy behaviour when attempting to parse incoming certificates with a wildcard. **Windows 8 and 8.1** are known to choke on wildcard certificates.
Signature algorithm	Minimum: SHA-1 Recommended: SHA-256 or higher	Server certificates signed with the signature algorithm MD5 are considered invalid by many modern operating systems, e.g. ***Apple iOS 6.x and above***. Also **Windows 8.1** and all previous versions of **Windows (8, 7, Vista)** which are on current patch levels will not validate such certificates. Having a server certificate (or an intermediate CA certificate) with MD5 signature will create problems on these operating systems. Apparently, no operating system as of yet has an issue with the root CA being self-signed with MD5. This may change at any point in the future though, so when creating a new CA infrastructure, be sure not to use MD5 as signature algorithm anywhere. The continued use of SHA-1 as a signature algorithm is not recommended, because several operating systems and browser vendors already have a deprecation policy for SHA-1 support. While the deprecation in browser-

		based scenarios does not have an immediate impact on EAP server usage, it is possible that system libraries and operating system APIs will over time penalise the use of SHA-1. Therefore, for new certificates, SHA-256 is recommended to avoid problems with the certificate in the future.
Length of public key	Minimum: 1024 Bit Recommended: 2048 Bit or higher	Server certificates with a length of the public key below 1024 bit are considered invalid by some recent operating systems, e.g. ***Windows 7 and above***. Having a server certificate (or an intermediate CA certificate) with a too small public key will create problems on these operating systems. The continued use of 1024 bit length keys is not recommended, because several operating systems and browser vendors already have a deprecation policy for this key length. While the deprecation in browser-based scenarios does not have an immediate impact on EAP server usage, it is possible that system libraries and operating system APIs will over time penalise the use of short key lengths. Therefore, for new certificates, 2048 bit or more is recommended to avoid problems with the certificate in the future.
Extension: Extended Key Usage	TLS Web Server Authentication	Even though the certificate is used for EAP purposes, some popular operating systems (i.e. ***Windows XP and above***) require the certificate extension "TLS Web Server Authentication" (OID: 1.3.6.1.5.5.7.3.1) to be present. Having a server certificate without this extension will create problems on these operating systems.
Extension: CRL Distribution Point	HTTP/HTTPS URI pointing to a valid CRL	Few very recent operating systems require this extension to be present; otherwise, the certificate is considered invalid. Currently, ***Windows Phone 8*** is known to require this extension; ***Windows 8*** can be configured to require it. These operating systems appear to only require the extension to be present; they don't actually seem to download the CRL from that URL and check the certificate against it. One might be tempted to fill the certificate extension with a random garbage (or intranet-only) URL which does not actually yield a CRL; however this would make the certificate invalid for all operating systems which do evaluate the extension if present. So the URL should be a valid one.
Extension: BasicConstraint (critical)	CA:FALSE	Server certificates need to be marked as not being a CA. Omitting the BasicConstraint:CA totally is known to cause certificate validation to fail with **Mac OS X 10.8 (Mountain Lion)**; setting it to TRUE is a security issue in itself. Always set the BasicConstraint "CA" to false, and mark the extension as critical.

2.2.8 Consideration 3: Which certificates to send in the EAP exchange

End-user devices need to verify the server certificate. They do this by having a known set of trustworthy anchors, the "Trusted Root Certificates". These root certificates need to be available and activated on the device prior to starting the govroam login. Therefore, it does not serve any useful purpose to send the root CA certificate itself inside the RADIUS/EAP conversation. It is not harmful to send it anyway though, except that it unnecessarily inflates the data exchange, which means more round-trips during govroam authentication, and in turn a slower login experience. One possible exception is: there are reports of certain Blackberry devices for which it is advantageous to send the root CA certificate nonetheless; if you expect you need/want to support Blackberry devices, sending the root CA may be of help.

During the EAP conversation, the govroam IdP RADIUS server always needs to send its server certificate.

One question needs an administrative decision: if there is one or more intermediate Cas between the root CA and the server certificate (such as is the case with, for example, the TERENA Certificate Service (TCS) and many commercial Cas), should the intermediate CA certificates be sent to the end user device during the EAP conversation, or should the devices pre-install the intermediate Cas along with the root certificate?

In any case, for a successful verification of the server certificate, the end-user's device must have the full set of CA certificates available. It does not matter whether the intermediate Cas have been pre-provisioned or are sent during the login phase; but if any one intermediate CA is missing, the verification of the server certificate will fail.

Pre-provisioning the intermediate Cas has the advantage of a relatively small amount of data being sent during the EAP authentication, which means fewer round-trips between the end-user's device and the govroam IdP RADIUS server. The downsides of this approach are that any changes to intermediate Cas (re-issue, rollover) will also need to be pushed to end-user devices. Also, if end-user devices are not under administrative control of the IdP, there is a danger that some end users do not follow the advice to install all intermediate Cas even though they should, and end up in a situation where the server certificate cannot be validated.

Sending the intermediate Cas during the login phase means a longer time to authenticate due to more round-trips, but means that it is sufficient for client devices to install the root CA certificate; if intermediate Cas change, the new ones will always become available to the device during the next authentication data exchange.

For most deployments, it probably makes more sense to include the intermediate CA certificates during the RADIUS/EAP conversation.

3 Set up of several popular RADIUS servers

Although some of the examples are based on previous versions, they give a good impression on how to set up some commonly used RADIUS servers.

3.1 FreeRADIUS

Please refer to the eduroam wiki (https://wiki.geant.org/pages/viewpage.action?pageId=121346259) for an excellent overview on the configuration of FreeRADIUS.

3.2 FreeRADIUS with radsec

To encrypt all RADIUS traffic, radsec can be used. It requires an additional virtual server entry in the configuration, that defines the certificate to be used:

```
listen {
        ipaddr = *
        port = 2083
        type = auth+acct
        proto = tcp

        clients = radsec

    virtual_server = default

        limit {
                max_connections = 64
                lifetime = 0
                idle_timeout = 300
        }

        tls {
                private_key_file = ${certdir}/radsec.pem
                certificate_file = ${certdir}/radsec.pem
                dh_file = ${certdir}/dh
                random_file = /dev/urandom
                fragment_size = 8192
                ca_path = ${cadir}
                cipher_list = "DEFAULT"
                require_client_cert = yes
        }
}
```

and dedicated clients in the configuration:

```
clients radsec {
    client default {
        ipaddr = 0.0.0.0/0
        proto = tls
        secret = radsec
        org = RADSEC
        limit {
            max_connections = 64
            lifetime = 0
            idle_timeout = 300
        }
    }
}
```

3.3 Radiator

One popular RADIUS server is Open Systems Consultant's "Radiator. This section details its configuration.

Because of the EAP authentication within RADIUS, a (small) PKI is required. If there is no PKI available, you could create the required key and certificate with, for instance, TinyCA. TinyCA (http://tinyca.sm-zone.net/) is a simple graphical interface on top of OpenSSL. It is possible to use OpenSSL directly (but instructions to do so are outside the scope of this document).

Depending on the EAP-type used, client certificates may also be needed.

Within the Radiator distribution there are also simple scripts available to create certificates for testing purposes.

The Radiator RADIUS server needs the configuration file /etc/radiator/radius.cfg.

This configuration file can be created with the editor of choice, for example

```
vi /etc/radiator/radius.cfg
or
pico /etc/radiator/radius.cfg
```

In the following examples there are two kinds of EAP that are configured at "institution":

- EAP-TLS based on client-certificates.
- EAP-TTLS and EAP-PEAP that do not require client certificates but use the traditional mechanism of
  username/password authentication instead.

3.3.1 Clients

RADIUS is based on a client-server model. The NAS-devices (Access Points, switches etc.) forward credentials to a RADIUS server, i.e. act as a client, and therefore need to be defined on the RADIUS server. Other RADIUS servers can act as a client as well, so every kind of RADIUS-request can be forwarded to another server.

The clients are configured within Radiator using the <Client>-clause:

```
<Client 192.168.10.200>
            Secret 6.6obaFkm&RNs666
            Identifier ACCESSPOINT1
            IdenticalClients 192.168.10.201
</Client>
```

In this example there is a client definition for 192.168.10.200, an Access-Point. The "secret" is a series of (at best 16) characters that are used to encrypt the credentials sent in the RADIUS-request.

It is of course recommended to create a secret that cannot be guessed easily, otherwise the RADIUS-message can be decrypted. This is not an issue with EAP-authentication using 802.1X, since the credentials are also transmitted over a SSL-encrypted tunnel between the client and the final authentication server. However, with regular credentials (like those used with Web-based redirection

authentication) this is sensitive information that might be captured, therefore a reasonably complex secret and an SSL tunnel is recommended.

The Identifier in the Client-definition can be used later on in the Radiator configuration to filter a specific request.

If more than one Client is to use this same secret and identifier definition, the IdenticalClients statement can be used. If there are many clients with different IP-addresses, there is also the possibility for a “catch-all” client. This is the default client that is used after all other client definitions didn’t match. Define this client as:

```
<Client DEFAULT>
```

If this kind of configuration is used, it is worth filtering with firewall-rules on RADIUS packets. There are only a few places where a RADIUS-request should come from; the management VLAN with the NAS-devices (switches and access-points), and the RADIUS infrastructure where unknown requests can be sent to.

3.3.2 Realms and VLAN assignment

The processing of authentication and accounting requests is done by linear processing of the present <Realm>- or <Handler>-clauses in the Radiator configuration file. Handler-clauses are more potent than Realm clauses in terms of filtering anything besides realms, and are therefore the preferred method. A realm is the part behind a username to indicate the origin of a user. With RADIUS, the username is usually separated from the realm with a “@” so the complete username looks like a regular e-mail address.

A <Handler>-clause is terminated with a </Handler>.

Within a Handler many mechanisms can be configured that define what to do with the RADIUS request.

3.3.3 PROXY example

The simplest Handler for proxying the request to another server uses the “AuthBy RADIUS” definition within this Handler.

In this example a proxy-configuration is shown. First we have a Handler that matches on any request, as long as it does not come from the client with the identifier “Proxy-Identifier”. This is to prevent a proxy loop. When a request comes from a proxy-server, it should never be forwarded back to that proxy-server.

Another important part is the hostname to which the request should be forwarded. Multiple hostnames can be defined here for redundancy reasons: if the first host does not respond within three seconds, the second one is tried instead. The UDP ports to which the RADIUS-request should be forwarded can be defined in this “AuthBy RADIUS” clause as well.

```
<Handler Client-Identifier=/^(?!Proxy-Identifier$)/>
   <AuthBy RADIUS>
      Host             192.87.36.3
      Secret          super_secret!
                     AuthPort     1812
```

```
                    AcctPort     1813
      StripFromReply Tunnel-Type, Tunnel-Medium-Type, Tunnel-Private-Group-ID
      AddToReply Tunnel-Type=1:VLAN, Tunnel-Medium-Type=1:Ether_802,
                                                    Tunnel-Private-Group-ID=1:909
   </AuthBy>
</Handler>
```

For a “Host”, both the IP-address and FQDN can be used. The choice is more or less a personal preference of the RADIUS administrator, but be aware that the hostnames are only looked up once at the Radiator (re)start. If the lookup fails, the Host cannot be used until the next restart. This can represent a problem at a power outage, where for instance the DNS server is not yet available even though Radiator is.

While by using hostnames one benefits from the administrative ease when an IP-address is changed, it is still necessary to restart the RADIUS server.

The last part in this <AuthBy RADIUS>-definition shows the addition of RADIUS-attributes to the RADIUS- response. These attributes can be used to define a VLAN that will be assigned to users that are authenticated using this Handler. With StripFromReply, the attributes that came from the proxy-server are stripped first to prevent malicious VLAN-assignments, afterwards the attributes are added with the proper values for the local network design. In this case, VLAN 909 is used for guests.

3.4 Secure authentication with EAP-TLS

EAP-TLS requires both server and client certificates. Rolling out such certificates is a sometimes daunting administrative process, and is out of the scope of this document. The remainder of this section assumes that client certificates have been issued to the users already.

In this example the AuthBy-definition is outside the Handler, and is referred to using the Identifier. (This is useful if the AuthBy-definition is reused in another Handler, for instance.)

```
<AuthBy  FILE>
     Identifier ID4-TLS
     Filename %D/TLS-users
     EAPType TLS
     EAPTLS_CAFile %D/cert/institution-ca-chain.pem
     EAPTLS_CertificateFile %D/cert/radius-server-cert.pem
     EAPTLS_CertificateType PEM
     EAPTLS_PrivateKeyFile %D/cert/radius-server-key.pem
     EAPTLS_PrivateKeyPassword (the secret that secures the private-key)
     EAPTLS_MaxFragmentSize 1024
     AutoMPPEKeys
     SSLeayTrace 1
     StripFromReply Tunnel-Type,Tunnel-Medium-Type,Tunnel-Private-Group-ID
     AddToReply Tunnel-Type=1:VLAN,Tunnel-Medium-Type=1:Ether_802,
                                        Tunnel- Private-Group-ID=1:909,User-Name=%u
</AuthBy>
```

In this AuthBy-clause there is an EAPTLS file defined that lists every employee. In this way, the users that can access the infrastructure using EAP-TLS are controlled.

The definitions that follow determine what to do with the EAP-request. First the “EAPType TLS” limits the use of this AuthBy-definition for TLS-only. Here regular password authentication is not desired, just certificates. Next, the certificate files are configured and the secret that secures the private-key file can be provided. If there is no secret for the private key, this can be omitted.

The next part defines in what size blocks the EAP-messages should be fragmented. Since some parts of the EAP-TLS challenge are too big to fit in a RADIUS request, the packets should be fragmented.

The MPPE-keys (Microsoft Point to Point Encryption, the protocol for encrypting the data across links) portion is important for wireless access. With 802.1X, encryption occurs at the edge of the network, between the Access- Point and the client. To provide this secure encryption, a unique key is created and encrypted using the MPPE- keys that are derived from the SSL-challenge. This can be done at the Access-Point and the Client end so that there is no need to transfer the WEP-key in plain text over the air. This, and the fact that the key can be rotated within a period defined by either the Access-Point or the RADIUS server, provides 802.1x users with a good level of security.

The last part of the AuthBy-definition shows how to assign a proper VLAN.

```
<Handler Realm=30unnelled30n.cc, EAP-Message=/.+/>
     AuthBy   ID4-TLS
</Handler>
```

The Handler above shows the referral to the AuthBy-definition and some filtering mechanisms to filter out the proper requests. If more things need to be filtered on, they can be added to this handler, as follows:

```
NAS-Port-Type=/^(Wireless-IEEE-802-11|Ethernet)$/
```

In this way, only requests with the proper NAS-Port-Types are allowed. For Accounting purposes, a new handler should be defined in this case, that filters on:

```
"Request-Type=Accounting-Request"
```

since the request does not match the Handler that filters on the EAP-Message.

3.5 EAP-TTLS or EAP-PEAP

When issuing end user certificates is not an option, the EAP-mechanisms PEAP and TTLS can be used.

These two mechanisms look the same in that they both set up a TLS tunnel on which the credentials can be transported. They vary in the supported password encryption schemes.

Virtually all implementations of PEAP encrypt the user's password as an NT hash exclusively. TTLS implementations typically offer plain text transport of the password, called TTLS-PAP (the outer TLS tunnels makes sure the password cannot be eavesdropped) and sometimes other encryption schemes like MS- CHAPv2.

Administratively, the choice whether to use PEAP or TTLS can be challenging.

Technically, three backend cases need to be considered for deployment:

Backend stores passwords in...	PEAP-MSCHAPv2?	TTLS?
Plain text or reversibly encrypted	Yes	Yes (TTLS-PAP, TTLS-MSCHAPv2)
NT-Hash	Yes	Yes (TTLS-PAP, TTLS-MSCHAPv2)
other irreversible encryption	No	Yes (TTLS-PAP)

Where both options are possible, we suggest the following order of preference: TTLS-MSCHAPv2, PEAP- MSCHAPv2, TTLS-PAP (in descending order of preference).

Instead of a flat file, a more flexible backend for user accounts is a database like MySQL, or LDAP.

```
<Handler TunnelledByPEAP=1, Realm=tunnelled.institution.cc>
         <AuthBy FILE>
                   Filename %D/peap-users
                   EAPType MSCHAP-V2
         </AuthBy>
</Handler>

<Handler TunnelledByTTLS=1, Realm=31unnelled.institution.cc>
         <AuthBy FILE>
                   Filename %D/ttls-users
         </AuthBy>
</Handler>
```

In these Handlers, the filtering options “TunneledByPEAP” and “TunnelledByTTLS” define that the tunnelled authentication (with the username and password in it) is handled here.

The "outer authentication", where the SSL“tunnel is set up, looks like the TLS handler.

```
<Handler Realm=group_1>
          <AuthBy FILE>
                    Filename %D/users
                    EAPType TTLS, PEAP
                    EAPTLS_CAFile %D/root.pem
                    EAPTLS_CertificateFile %D/server.pem
                    EAPTLS_CertificateType PEM
                    EAPTLS_PrivateKeyFile %D/server.pem
                    EAPTLS_PrivateKeyPassword serverkey
                    EAPTLS_MaxFragmentSize 1024
                    EAPAnonymous anonymous@group1
                    AutoMPPEKeys
           </AuthBy>
</Handler>
```

3.6 Microsoft NPS

The following example govroam example setup is an adaptation form:
“Running eduroam on NPS with Windows 2008 R2 Enterprise”
DRAFT version 2013-07-15
Author: Paul Dekkers
License: CC-BY

3.7 Running govroam on NPS with Windows 2008 R2 Enterprise

The network policy server is the RADIUS server as part of Windows server editions. These instructions assume a basic setup of an Active Directory.

As a quick-start / overview, the following topics are covered in more detail in this document:

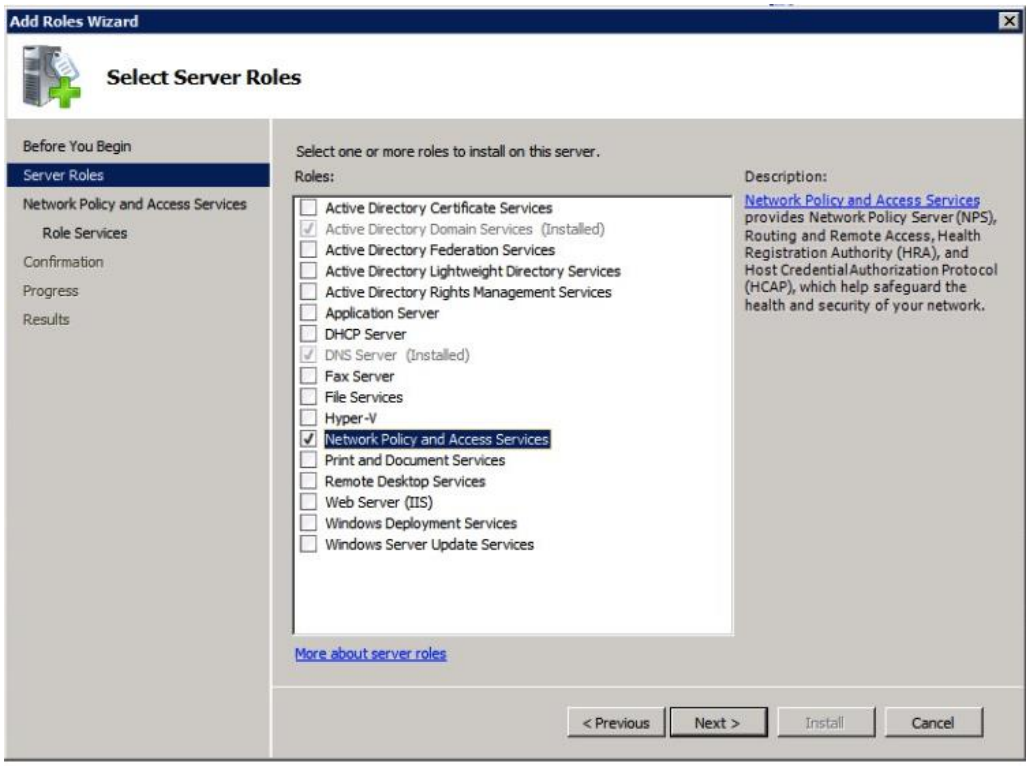
- Network Policy Server (NPS) needs to be installed as a server role;
- A server-certificate suitable for NPS (eg. One signed by a public CA) is required;
- You need to configure RADIUS clients (and shared secrets) in NPS to allow requests from your Access Points plus the proxy-servers (details provided and negotiated (shared secret) by your National Roaming Operator (NRO));
- The proxy-servers of your NRO will be configured in a RADIUS server group, with one server preferred (lower priority) and a secondary for failover (higher priority);
- You will have one or more Network Policies: these will handle the actual EAP authentication of your users. A policy can be duplicated to add VLAN-assignment attributes for local use, while remote users should not receive these attributes. Configure “Microsoft PEAP” for your policies (Add, then Edit to select the server-certificate) and deselect all “less secure” mechanisms.
- The Connection Request Policies determine how a request is dealt with: proxied to your NRO servers, or handled locally. For local-accounts, create a condition that matches your users with their realm, while preventing usage for unknown (sub)realms or no-realms.
    - A good regular expression for local users is for example “@lab\.gvoram\.nl$” as an explicit realm match for the User Name;
    - A good catch-all expression to handle unknown sub-realms would be “\.lab\.govroam\.nl$” that should come as lowest prio policy-rule before the proxies.
    - The User Name condition for the Connection Request Policy to the proxy-servers can be something like “@.+\.[a-z]{2,6}$” to match only valid realms;

The Network Policy Server has the following limitations:

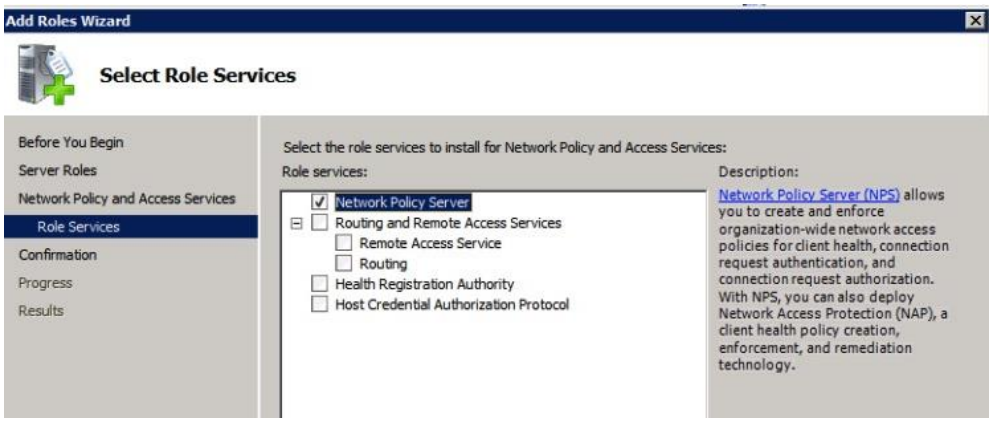
- You cannot strip attributes (for instance VLAN attributes assigned by other identity providers (IdPs)) but you can explicitly set values applicable to your environment if you work with VLANs or want to prevent invalid attributes;
- You cannot add attributes in outbound requests: adding an “Operator-Name” attribute to indicate where a user gets online is thus not possible and could be set by the National Roaming Operator instead;
- NPS doesn’t answer to Status-Server requests: it’s a best-practise for govroam proxy-servers to check your servers’ availability with those requests, and ideally you would do that the other way around too;
- Because of the previous limitations, inform your National Roaming Operator that you’re working with NPS;
- While the outer username (via the Connection Request Policy) can be rewritten, the inner username (often users configure both to be the same) handled by the Network Policy cannot. This means that your users will have to use the registered UPN (User-Principal-Name) which by convention maps to the e-mail address / user-ID@domain-name.

3.7.1 Installation of NPS

In the “initial configuration” or “Server Manager” look for “Roles” and click “Add Roles”.



Select the “Network Policy and Access Services” option and click “Next >”. After reading the introduction to NPS, continue to the role services to install:



Select only the “Network Policy Server” component, and click “Next >” again.

You will see a summary of the installer-actions, and need to click “Install” to continue. Wait for the installation to finish, and click “Close”.

You can now find the “Network Policy Server” under the “Administrative Tools” in the start menu, in the Server Manager, or as a snap-in to mmc.

3.7.2 Server certificate for NPS

You need to have a server certificate in order to use PEAP-authentication with govroam. PEAP sets up a secure tunnel (just like HTTPS does for websites) in order to protect the credentials, and is an

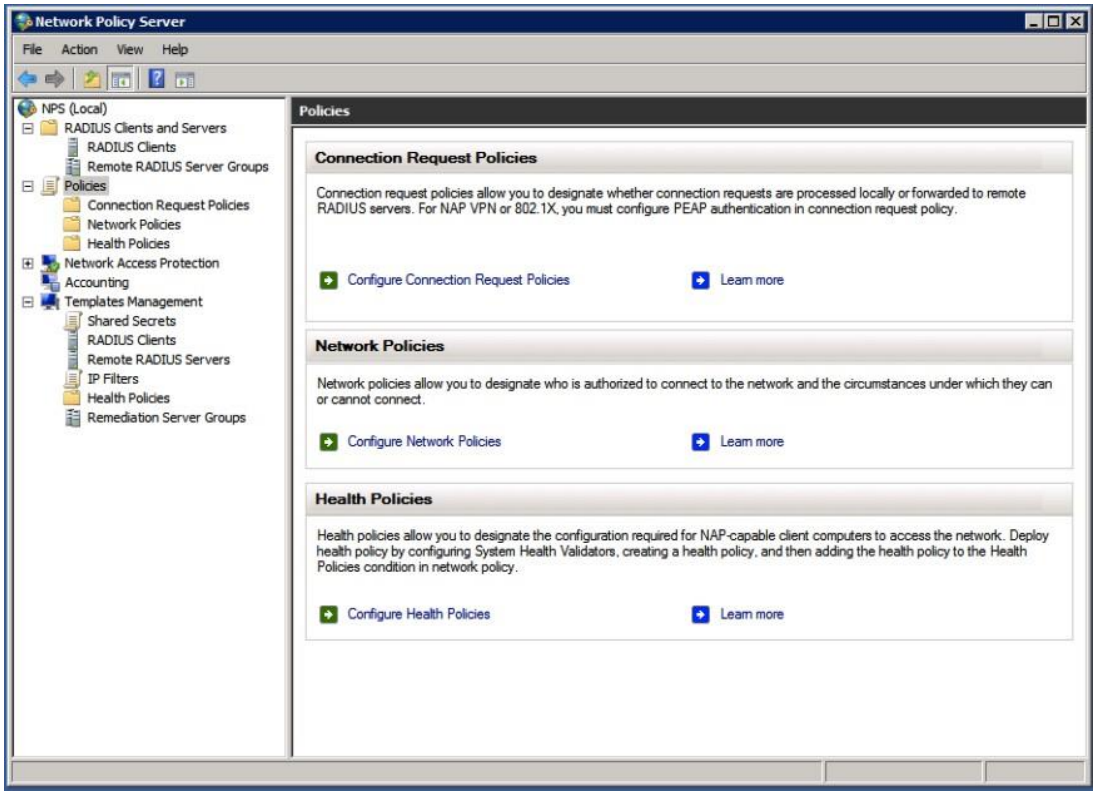
important part of the mutual authentication: both the user needs to prove who he is, and the authentication server needs to prove to the user that he or she is providing credentials to the right authority.

Without certificate (self signed or not) it's not possible to do local authentication. NPS can still be used as a proxy to receive requests from Access Points, log, filter, and forward to the govroam infrastructure.

If you have no certificate installed (or in doubt about your certificate), read Appendix A about Certificates.

3.7.3 Configuration of NPS

The NPS console (snap-in) allows you do use a Wizard for 802.1X / secure wireless. While you can use this for govroam, it doesn't provide all the required settings (like realm/user-name pattern-matching), so you need to make some more changes in the created policies anyway. In these instructions, we'll create the policies directly from the "Connection Request Policies" and the "Network Policies".



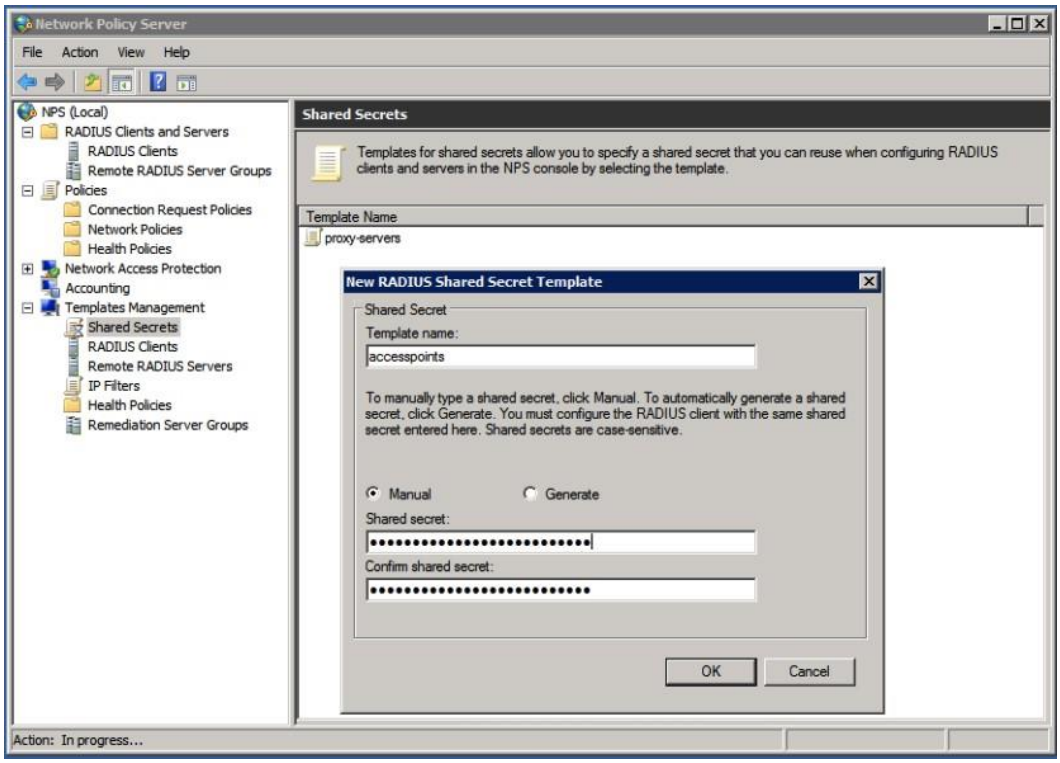
Before any policy can be applied to authentication requests, we need to create "RADIUS clients" in order to allow both your Access Points (and/or Switches) and the govroam infrastructure to actually send requests to your server (that's also a client).

To prevent typo's between multiple peers and allow easier changes, it's preferable to create a shared secret template for peers using the same shared secret. You can for instance create one for your

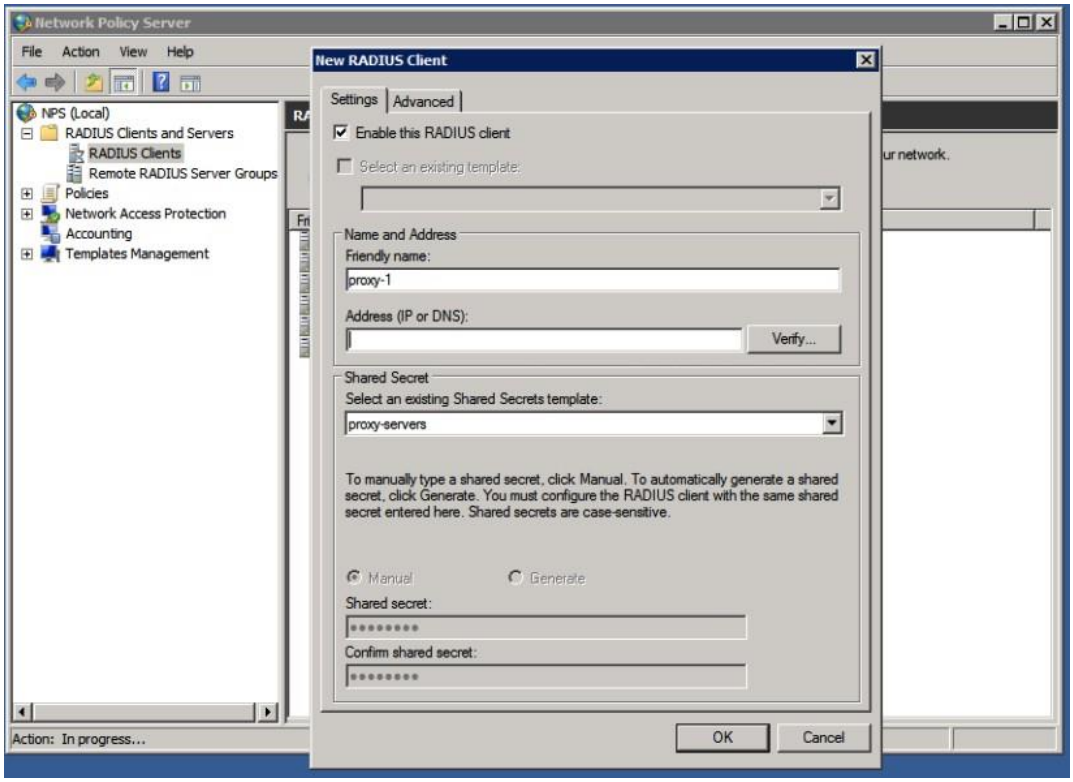
access points and one for the proxy-servers. The proxy-server secret you need to negotiate with your national govroam roaming-operator. The access-point secret, you configure

on your own access-points so you can make something up there yourself.

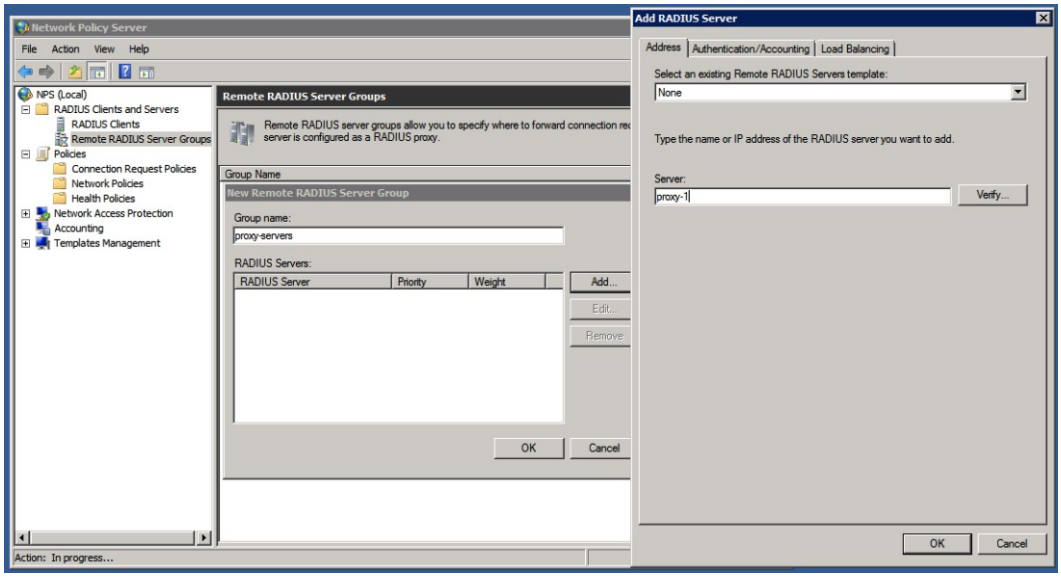
You can create these templates in the “Template Management” and “Shared Secrets” section, by right-clicking and selecting “New”...



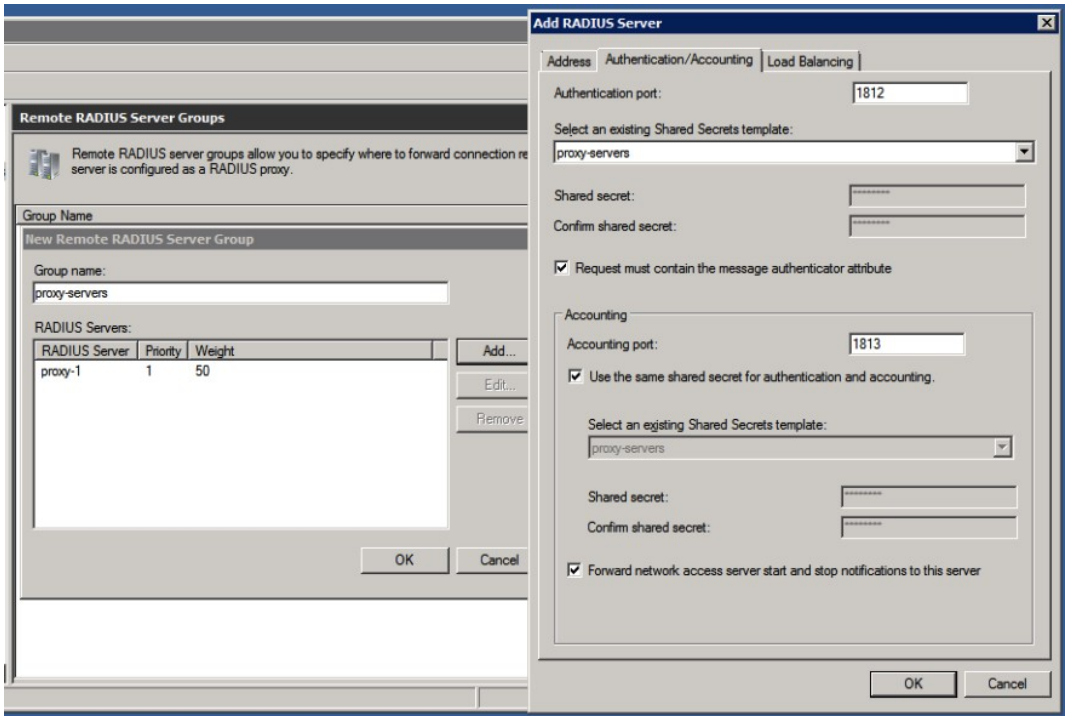
After creating the template, create clients for your access-points and proxy-servers, by right- clicking “RADIUS clients” (under RADIUS Clients and Servers) and “New”:



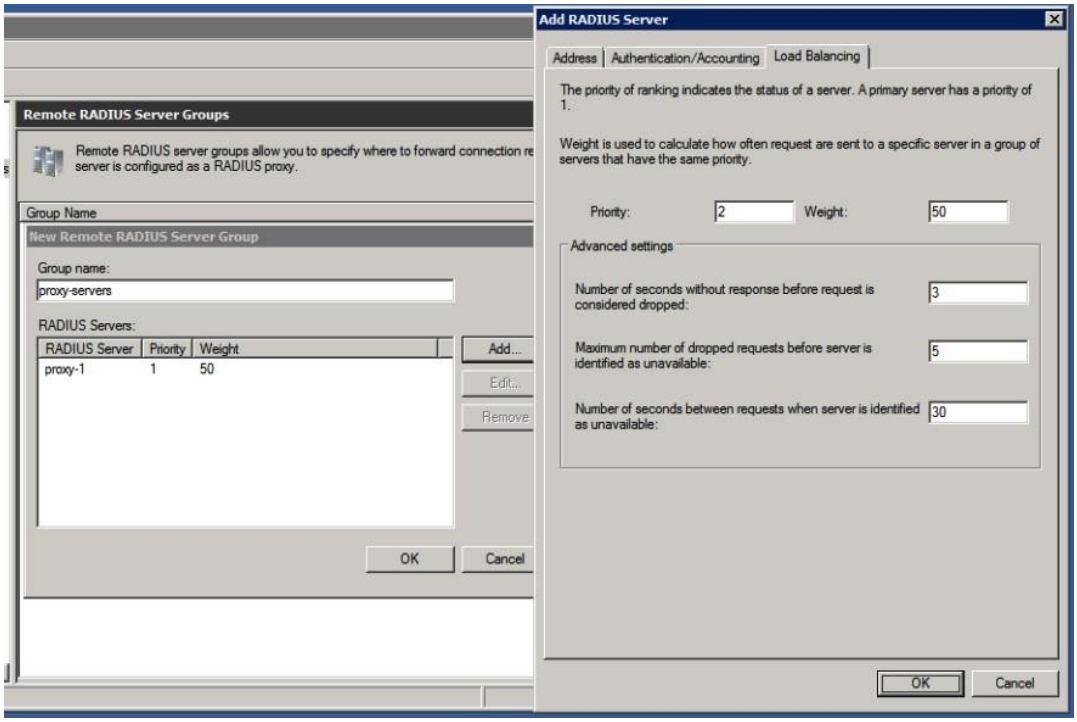
Now, we create a server group for the proxy-servers, that will be used to send authentication requests to for non-local users. In the “RADIUS Clients and Servers” right-click “Remote RADIUS Server Groups” and “New”...



Enter a name for your server group, such as “proxy-servers”, and click “Add...” to add one or more of the servers. Enter the proper name (proxy-1 in the example is not a proper name ;-) but you will get these details from your National Roaming Operator (NRO)), and proceed to the Authentication/Accounting tab for the shared secret settings:



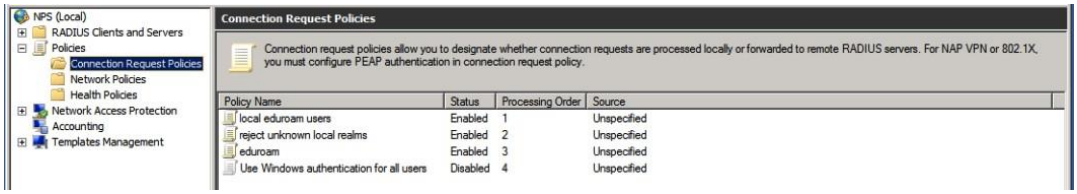
For a secondary server, consider the last tab “Load Balancing”. It’s recommended not to load-balance single EAP-sessions across multiple servers, which is what NPS will do when the Load-Balancing Priority is all set to the same level. In many situations it just works, but there’s no guarantee, so better set it to a lower priority so it’s only used for failover. If in doubt, ask your National Roaming Operator for advice.



The “Connection Request Policy” is there to decide what to do with an authentication request: forward it to a proxy-server, or authenticate locally. The decision is based on RADIUS attributes, such as the User-Name, but this can also be a RADIUS client IP-address or friendly-name for instance.

The order or Connection Request Policies is important. You can move policy-rules up and down, and also temporarily disable a rule.

A typical order is as follows:

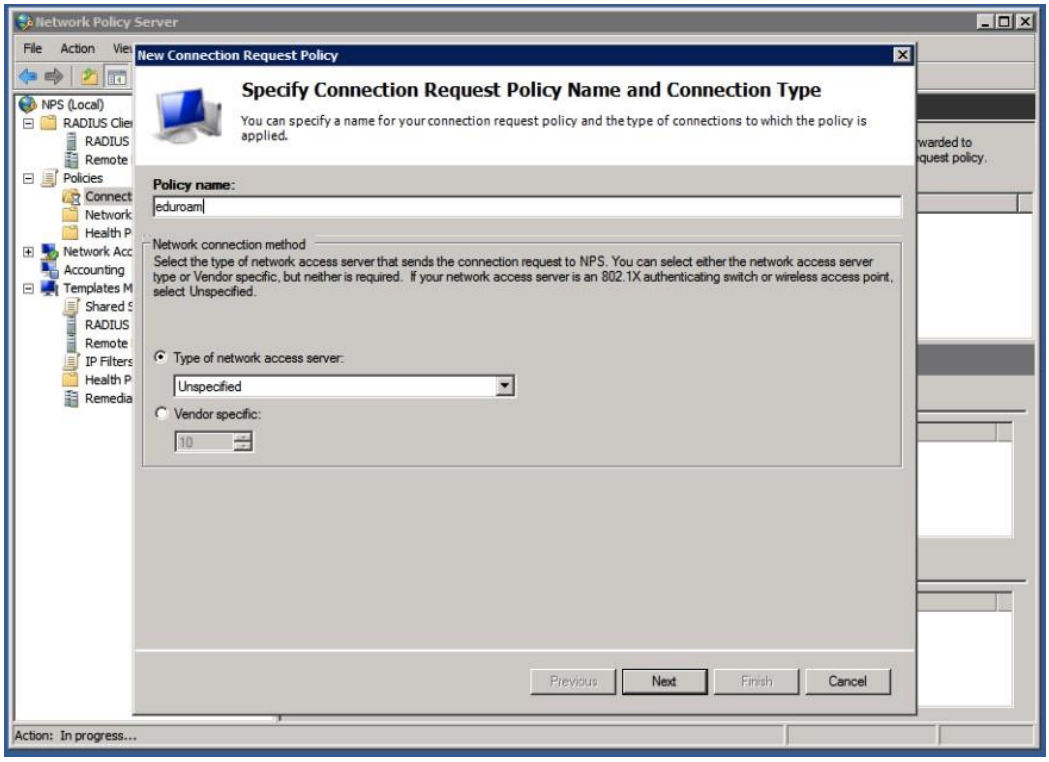


1. authenticate local users @your-realm.tld
   (you can add more for eg. @student.your-realm.tld)

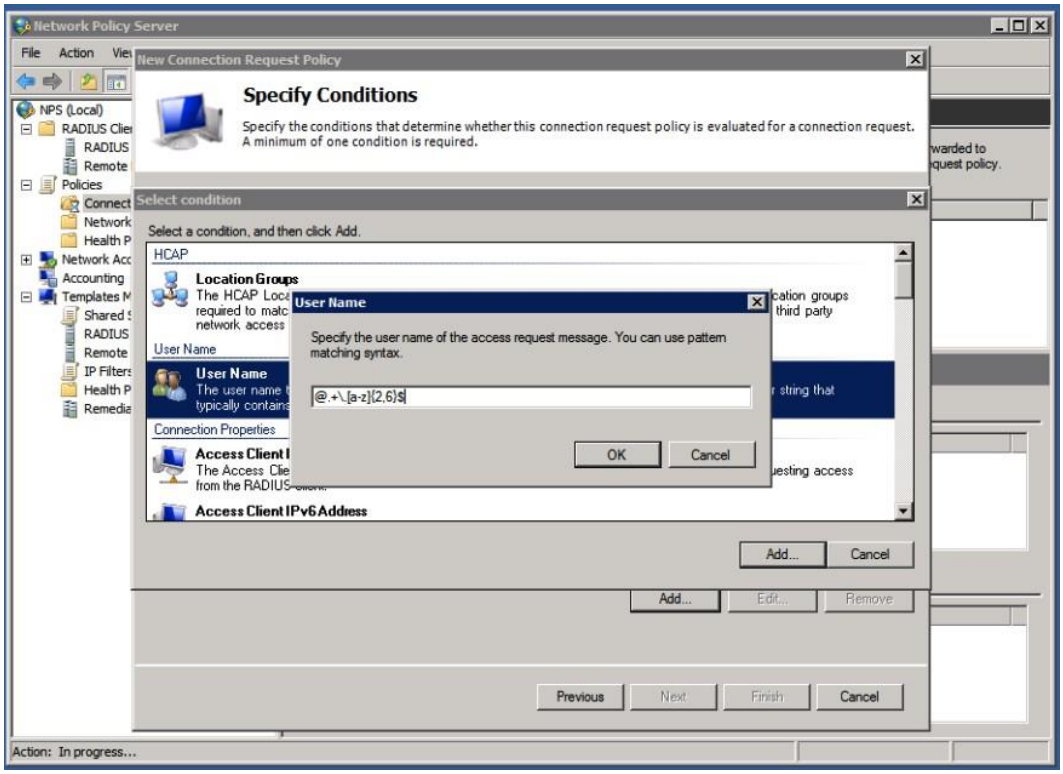
2. authenticate mis-matches in your-realm, such as non-existant.your-realm.tld

3. forward to remote proxy-servers

First, create a connection to the proxy-servers by right-clicking the “Connection Request Policies” and “New”.

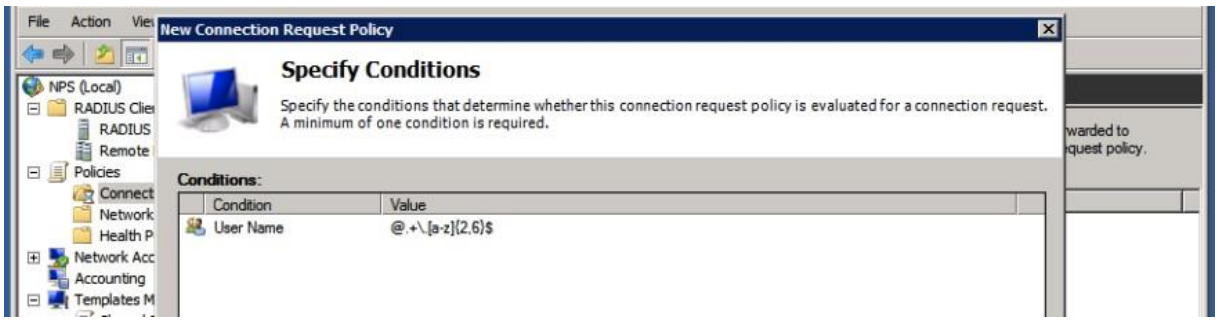


Give the policy a name (such as “govroam”), and click “Next”. Now conditions for matching this policy have to be specified. This rule will be based on User-Name matching.



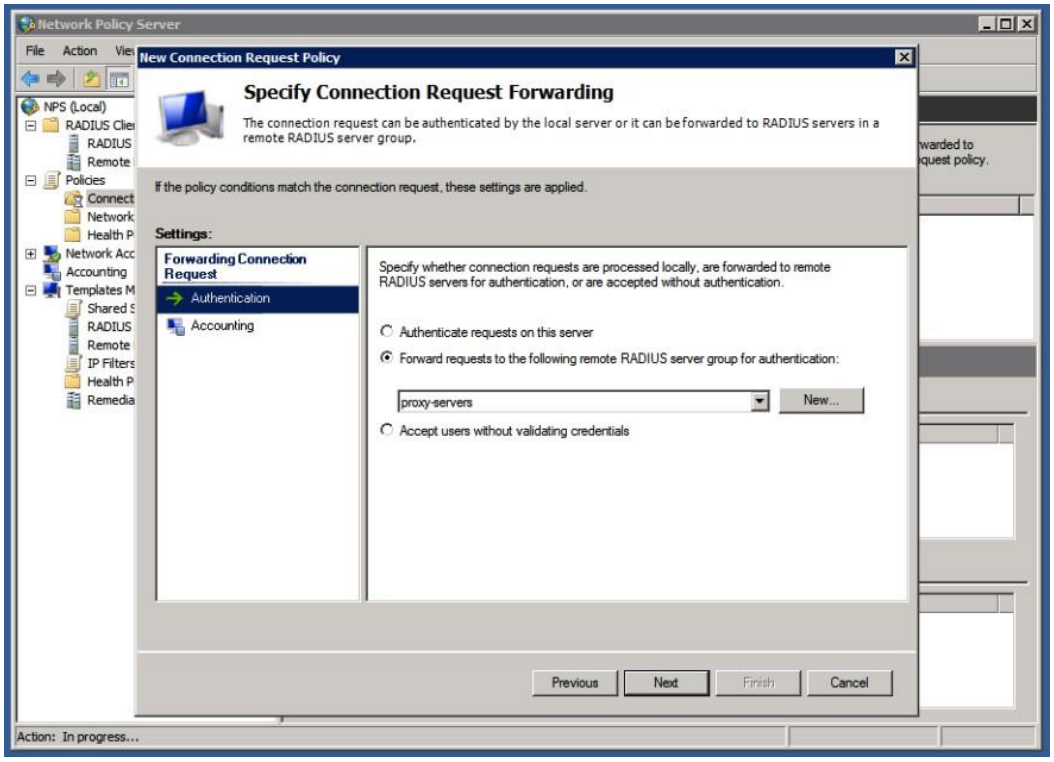
We use a regular expression here to match user-names that look valid. User-names in govroam are like the e-mail addresses, and end on something.tld (something.nl in case of Dutch government organisations) – that means we shouldn’t forward realms that have no dot “.” In them, or when there’s no realm (after the @-sign, the domain is realm in RADIUS-slang) at all.

The regular expression @.+\.[a-z]{2,6}$ is a case-insensitive match for realms ending on something dot tld between 2 and 6 letters. Keep in mind that this might change in the future when internationalized top-level domains are allowed, then this regular expression might need to be updated. A more lenient regexp would be @.+\..+$ to allow a realm with something dot something as a minimum. Both regexps handle any number of sub-realms.

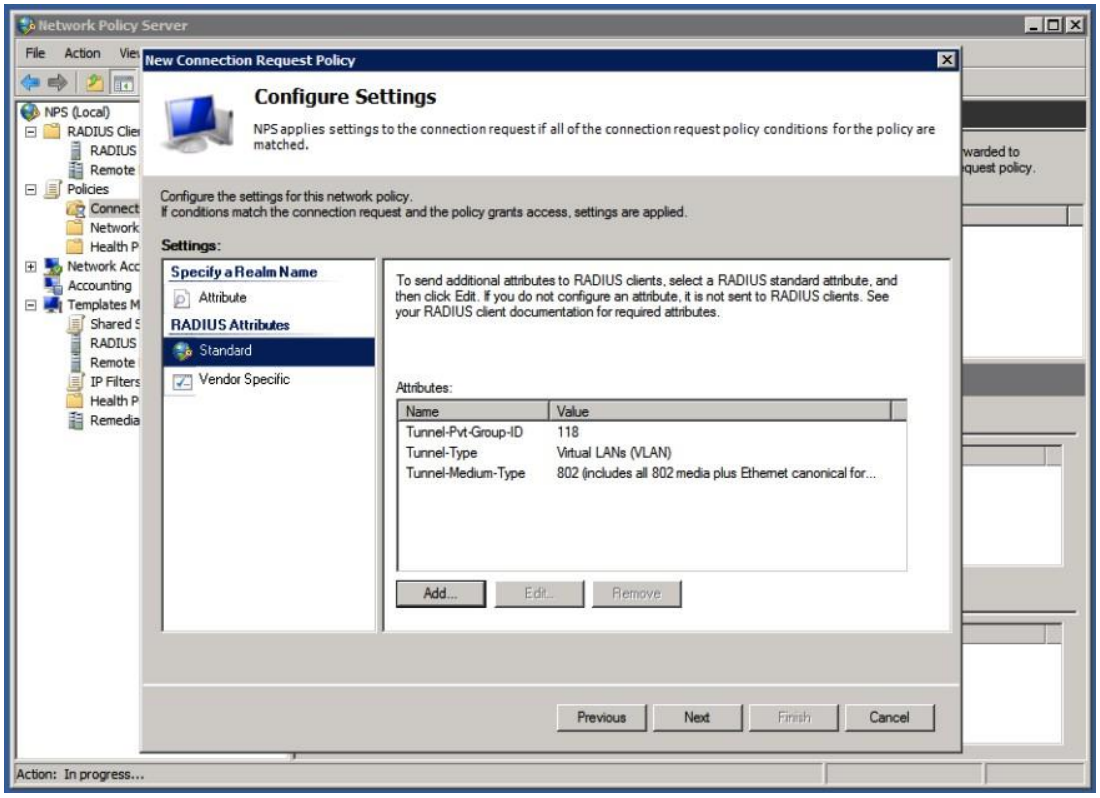


After the condition is set, click “Next”. (Note that the default Wizard also adds a condition for NAS Port Type, and sets this to “Wireless – IEEE 802.11”. This is fine if your Access Points add this and if you need to do that kind of filtering of RADIUS requests. You could add “Wired”, as some AP’s seem to be non-compliant and add this.)

Next, specify what to do with the requests that match the condition. In this case, we want to forward the request to the proxy-servers, so the RADIUS server group needs to be selected:

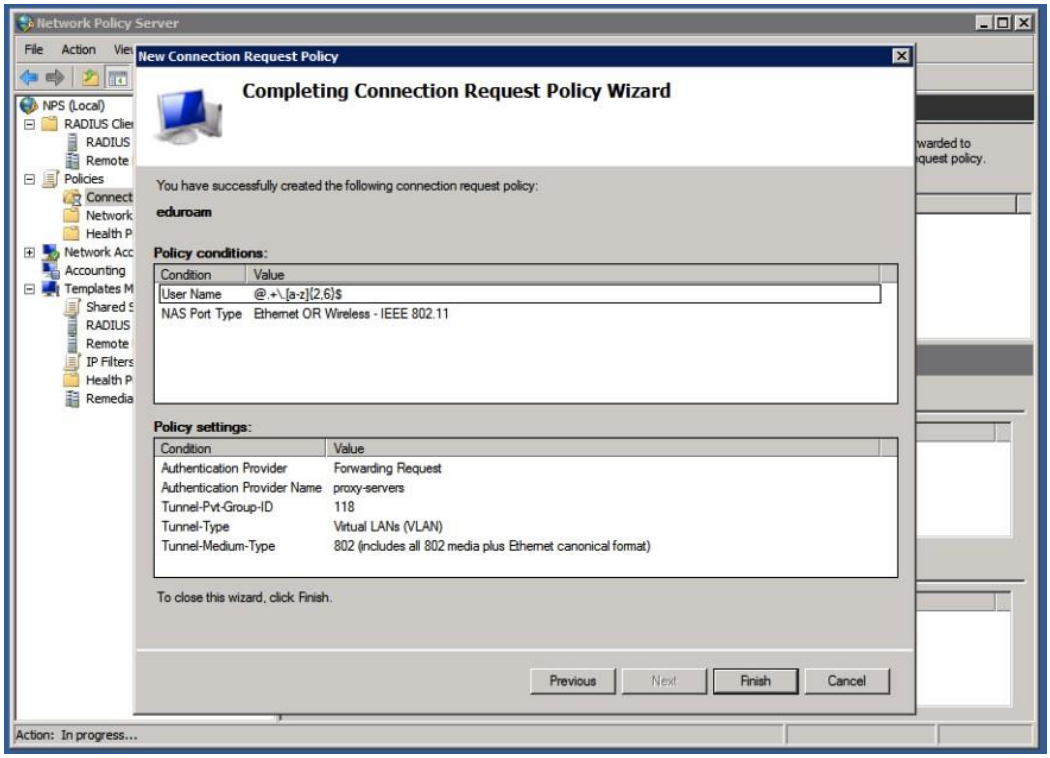


The final configuration options for a Connection Request Policy allows you to add RADIUS attributes to the RADIUS reply. You don't need to do anything with this, but you can define or override VLAN attributes if your Access Point is configured to use VLANs. This way you can define a different VLAN for guests compared to local users. (More about VLANs for local-users later.)



The above example adds VLAN 118 for guests authenticated via your Access-Points. Look at your Access-Point documentation to find the actual attributes you need to use, some use non-standardized attributes for this. When in doubt, start without any of these attributes.

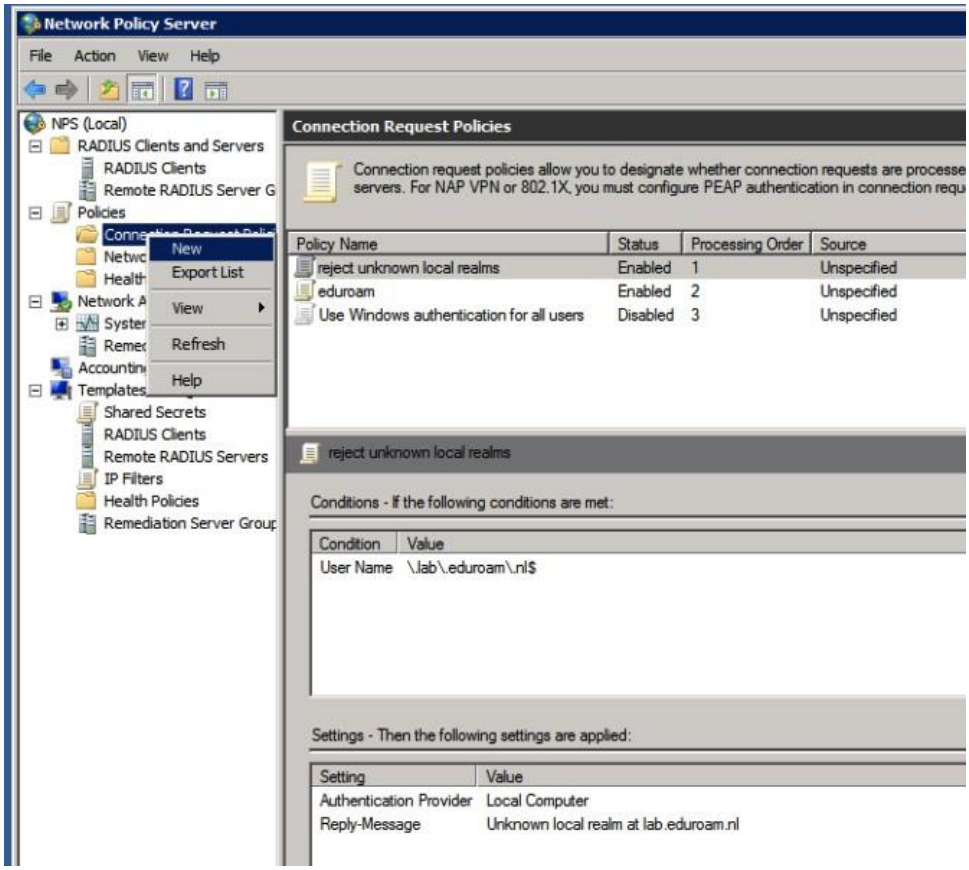
As final step, review the settings made by the wizard, and click Finish.



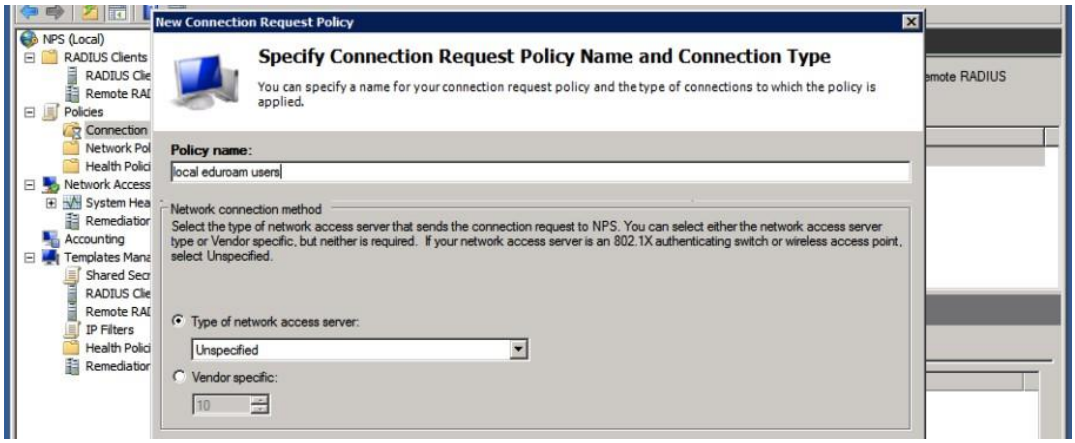
After creating a new policy-rule, always reconsider the order of policies. A policy might catch all the requests and make NPS not consider any of the newer requests.

While testing, take into account that it might take a second or two before NPS actually uses the newly configured settings.

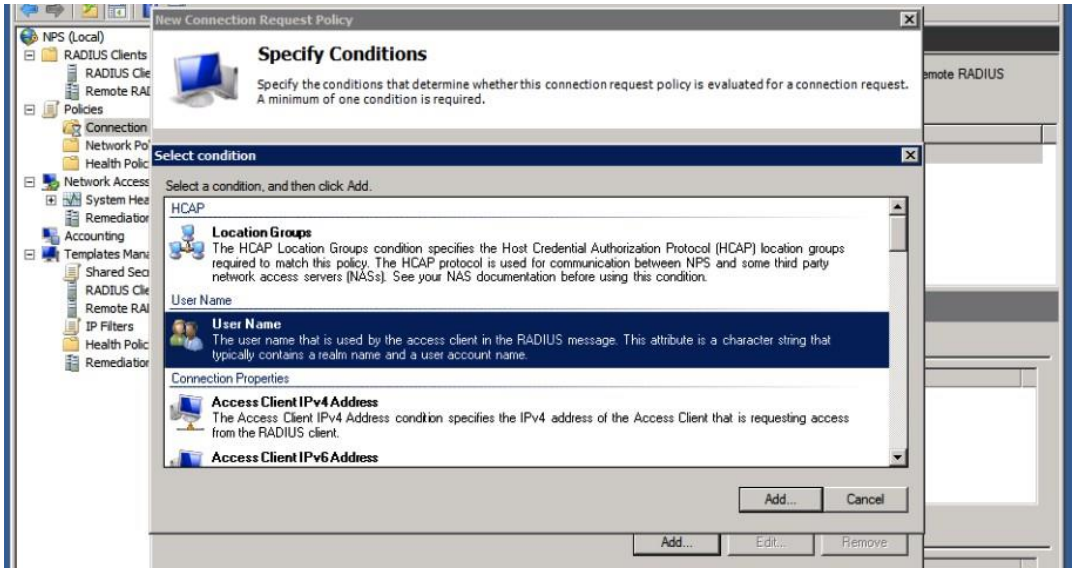
You could test govroam authentication with a remote (test)-account provided you have one. The next step is to create a policy for local users.



Again, click the “Connection Request Policies” tree-item and select “New”.



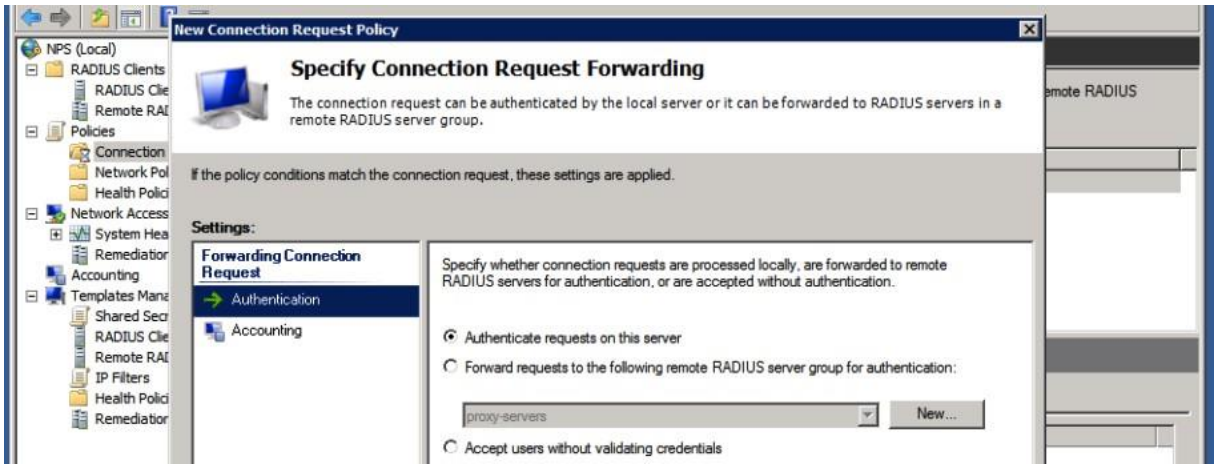
Give the policy a name such as “local govroam users”, and click “Next”.



Again, we need to specify conditions for the policy to match. In this case we'll want to match local users, by their user-names. This can be done by a regular expression:

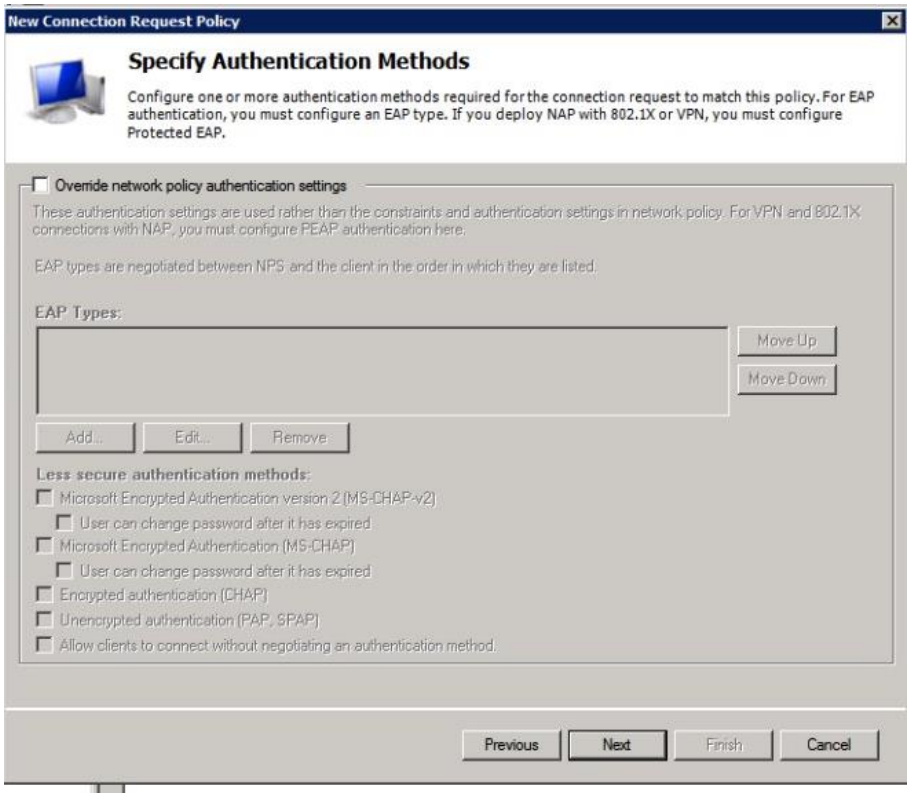
Please don't allow users to authenticate without realm! It will be very confusing for users if it works locally without @realm, and you need to have a @realm in a remote location. That breaks the whole working of govroam for this user and it will lead to misconfigured clients and support calls.

Click "OK" and "Next" when done.



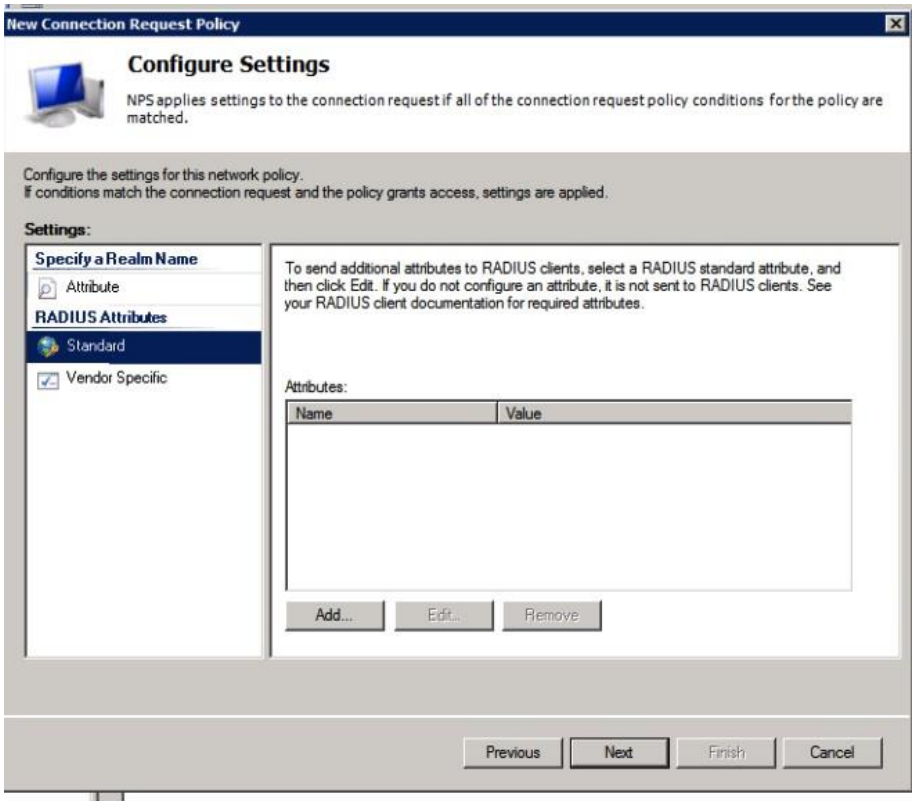
Now, select to "Authenticate requests on this server".

The next screen asks to override authentication methods configured for this user in the Network Policies.



Make sure no override is done.

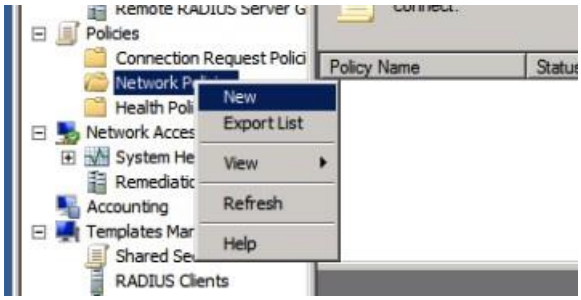
The next screen allows you to configure RADIUS attributes, but don't enter anything here.



If you want to assign VLAN attributes for your users, you'll need to do that in the Network Policy.

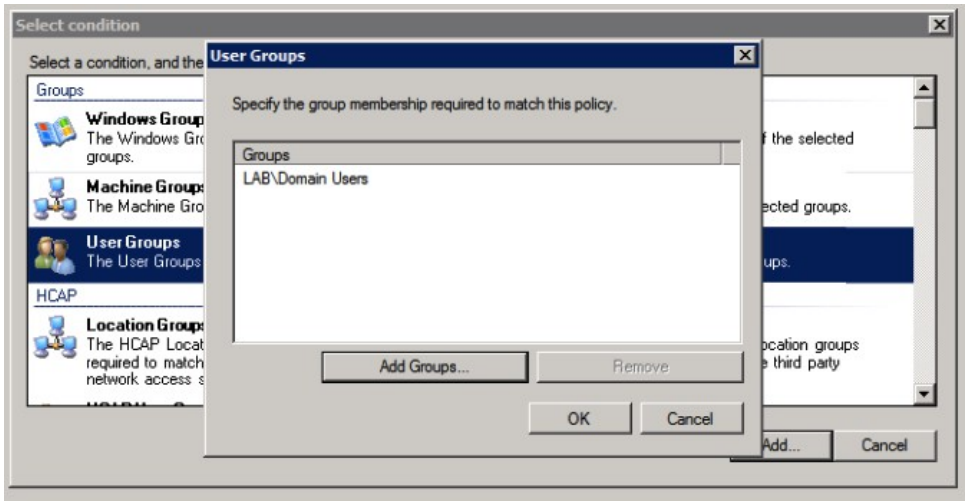
Review your settings, and click "Finish".

Next, create a Network Policy for your local users.

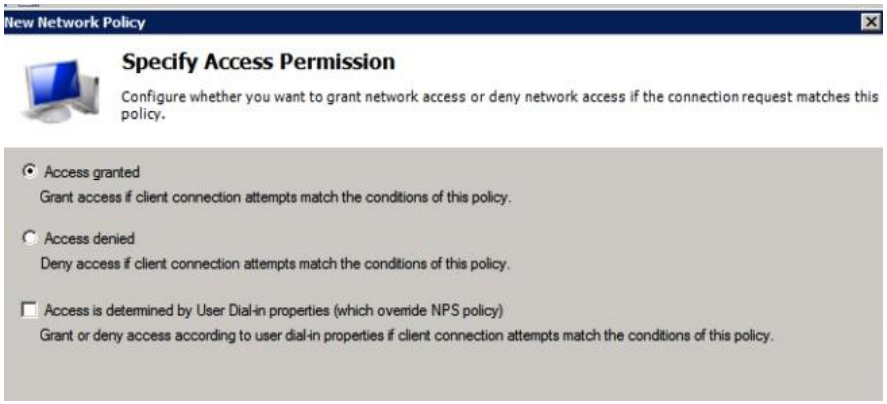


These policies are only used for Connection Request Policies that have "Authenticate requests on this server" set.

Give your policy a name such as "local govroam users" and leave the other settings default.

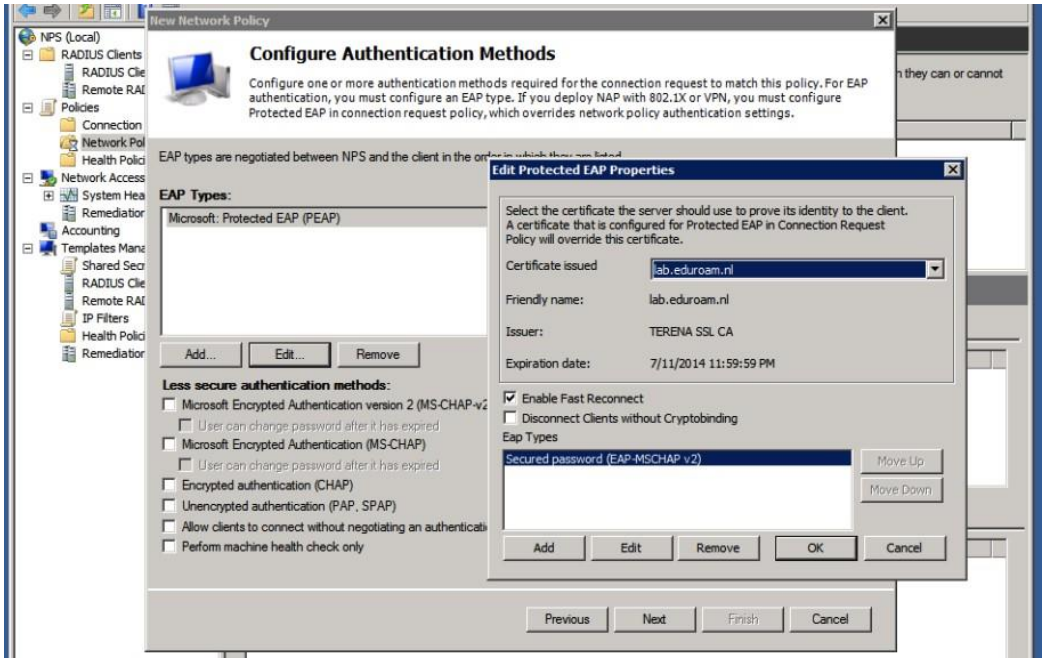


We need to specify the conditions for matching this request. Here you can define the users in your AD that are allowed to authenticate.

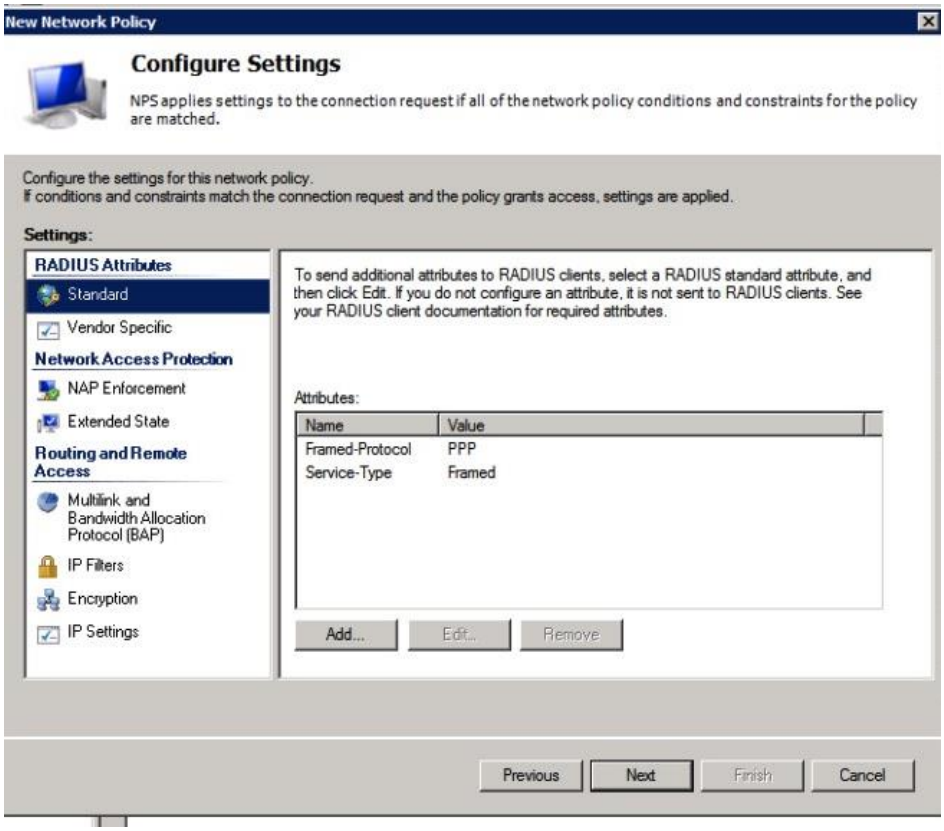


In the next screen, select to grant access to these users. Now for the authentication methods that are allowed:

Deselect the “Less secure authentication methods”, and click “Add...” to add an EAP type named “Microsoft: Protected EAP (PEAP)”.



Edit the PEAP settings, and make sure the proper certificate for the server authentication and TLS tunnel setup is selected. (See the Appendix about certificates if any of these steps give a warning or if you don't have a certificate installed just yet.)

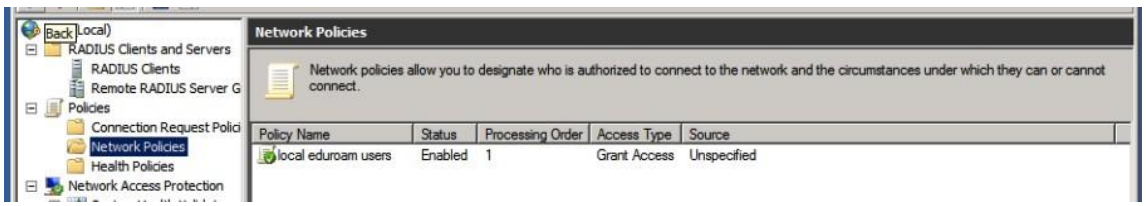


In the next step of the wizard you will have the chance to configure any RADIUS attributes.

But: don't add attributes just like that! If you want to override for instance the VLAN by setting attributes for your own users, you need to do this in a separate policy that only works for your local clients (Access-Points) only. If you set VLAN attributes for your users in authentication requests that originate from the govroam infrastructure your users might be denied access, which might be a difficult thing to debug.

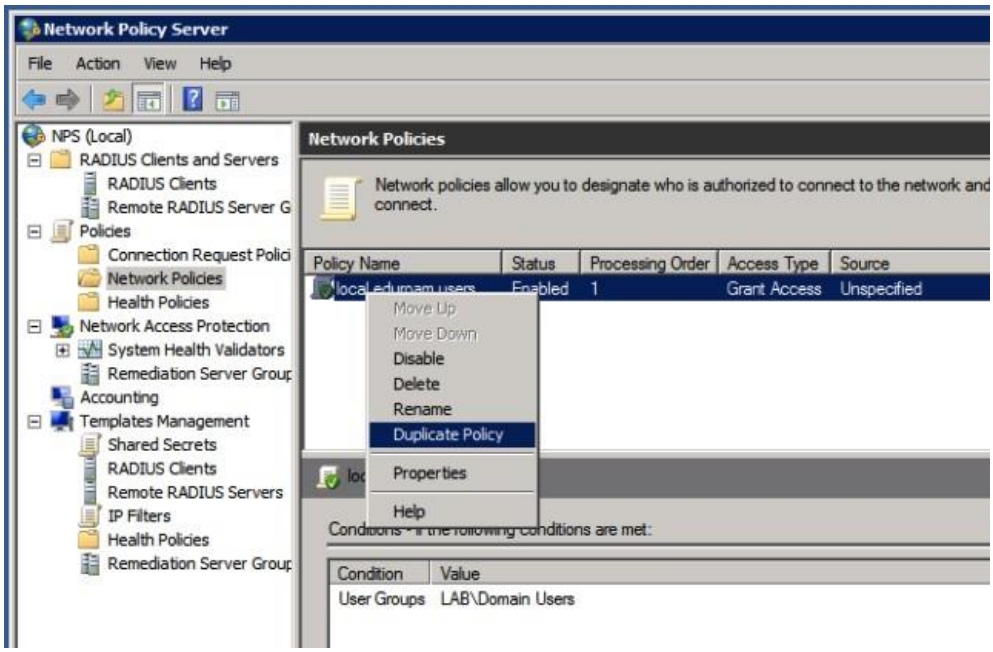
Don't use NAP enforcement or any of the other settings: they don't have value for govroam deployments.

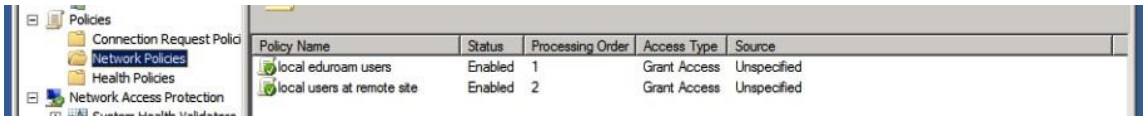
Finally, review your settings, and click "Finish".



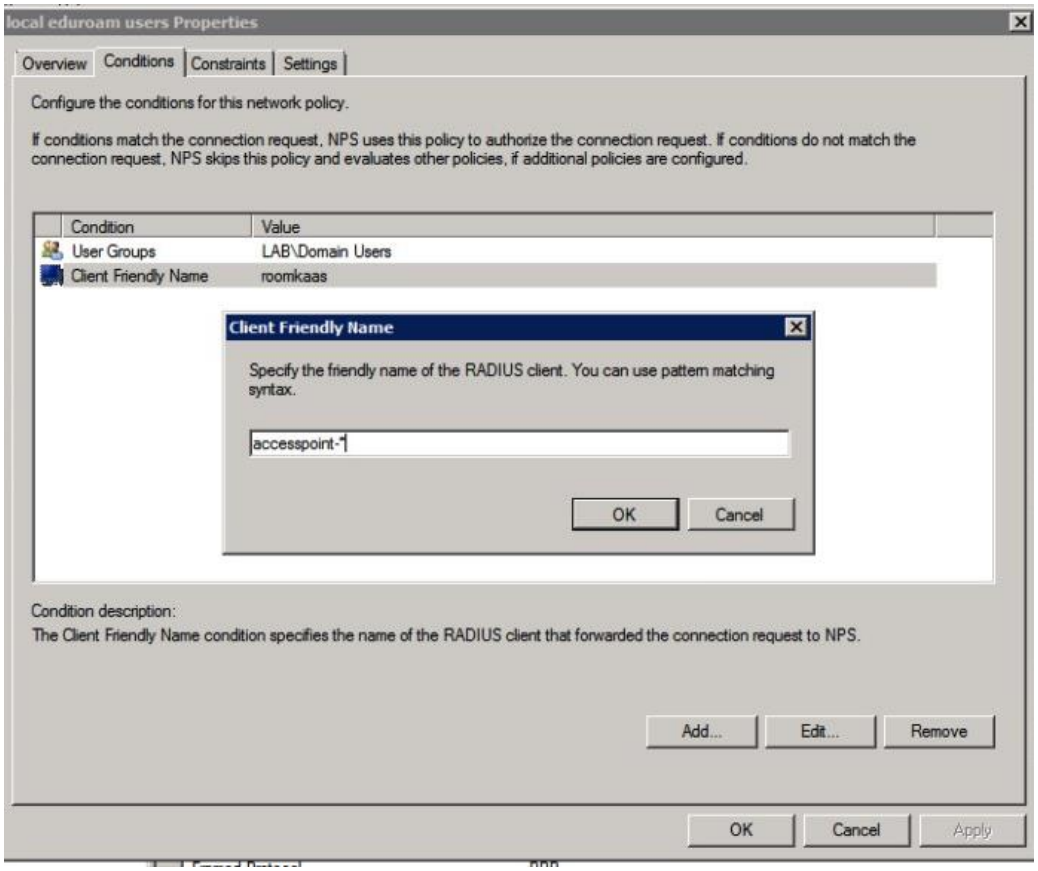
Your local accounts should now be able to authenticate wirelessly! Go ahead and try it, before making any more changes.

In order to assign VLAN attributes to your local users, we need to duplicate the Network Policy.



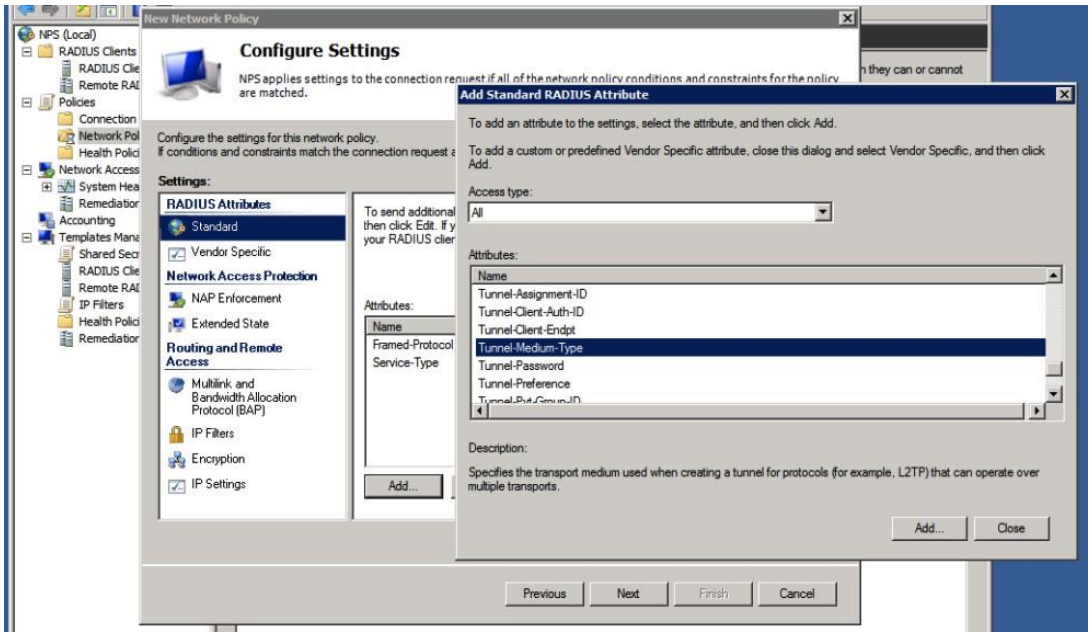


The order of rules is important: make sure the rule for matching local users is first. You can add extra conditions to this rule to make sure it only matches local requests, and add VLAN attributes in the properties (“Settings” tab) for this policy.

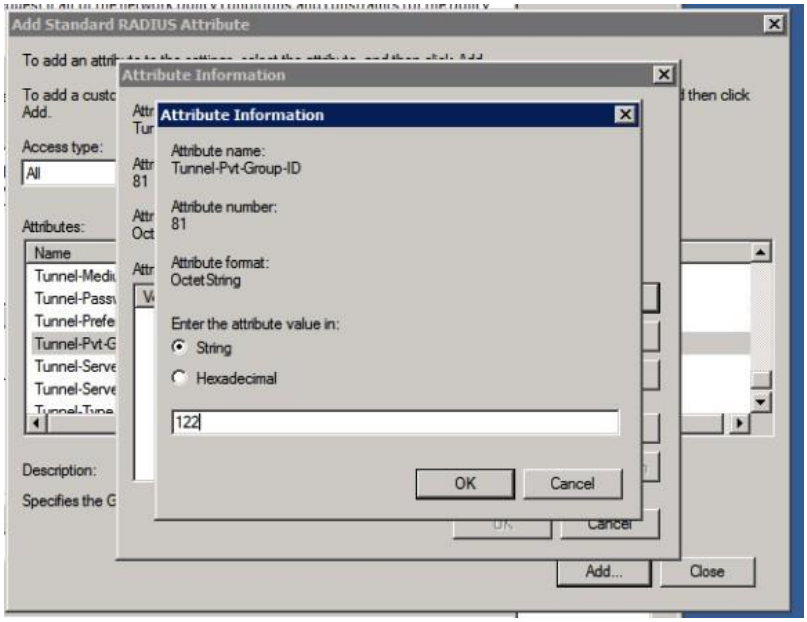


First, add a Condition to only match local requests. A simple example is to use the friendly name for your clients: if you named your clients accesspoint-1 and accesspoint-2, you can use an expression here like accesspoint-*

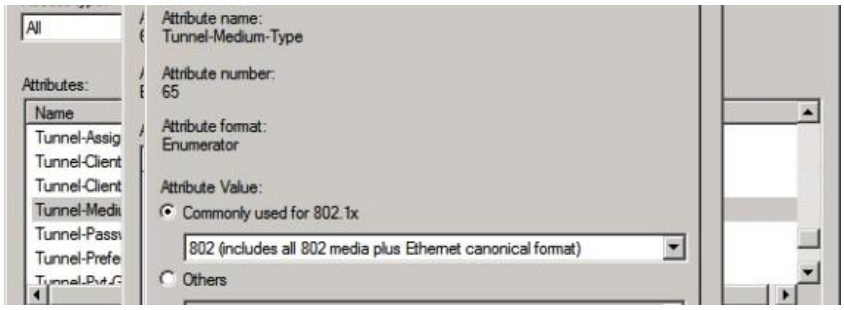
In the settings tab, add additional attributes for your users.



The standardized attributes for VLANs are Tunnel-Medium-Type, Tunnel-Type and Tunnel-Pvt-Group-ID where the Tunnel-Pvt-Group-ID contains the number of the VLAN you want to assign.

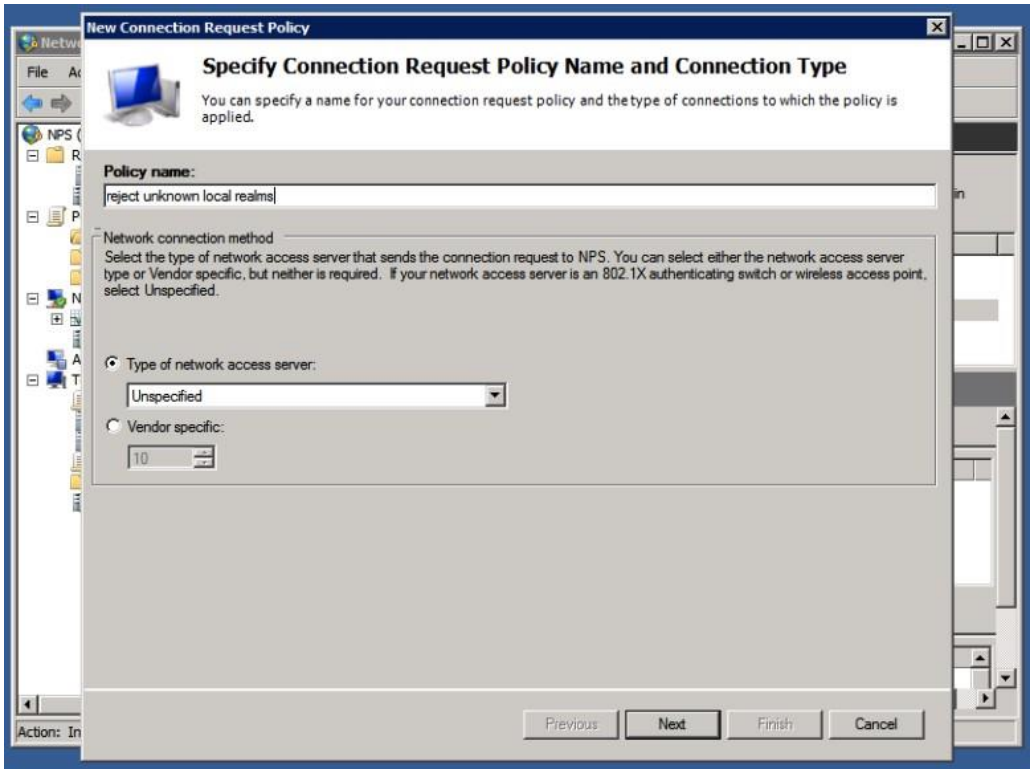


The other attributes need to contain default values,

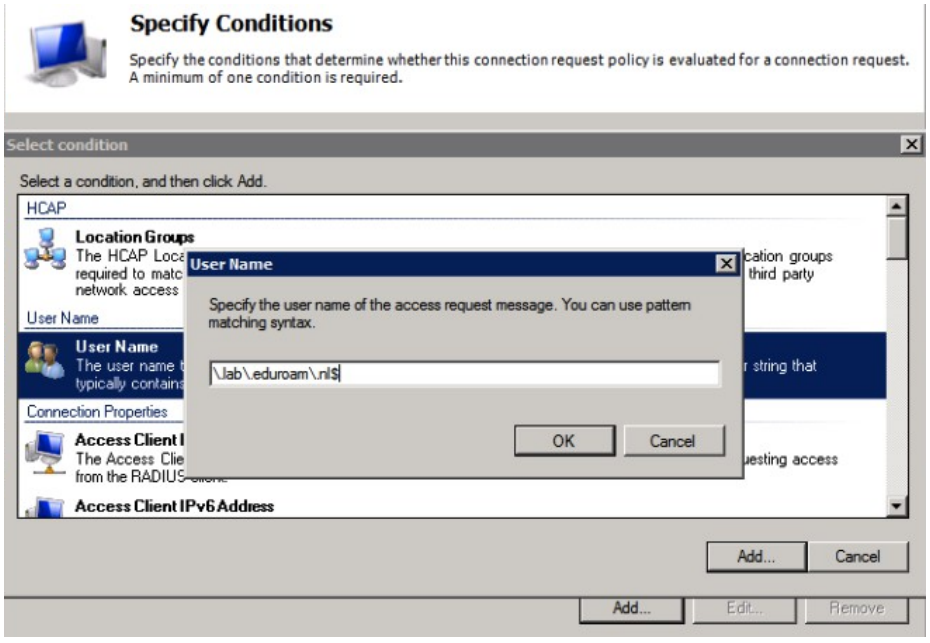


Tunnel-Medium-Type = 802, and Tunnel-Type = Virtual LANs (VLAN).

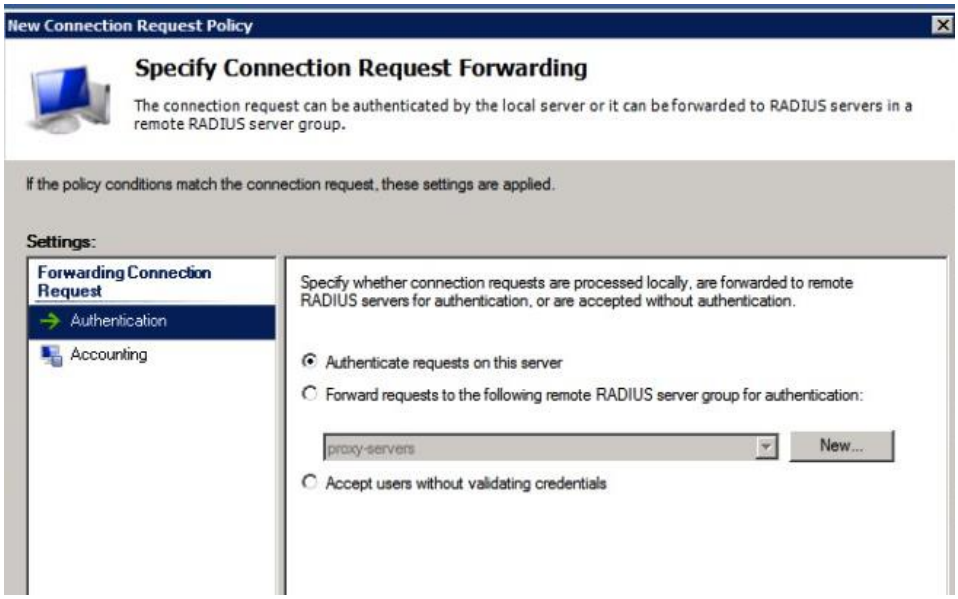
One last Connection Request Policy needs to be created (unless your National Roaming Operator only forwards the realms you're using to your servers).



We need to create a policy to reject “unknown local realms”: realms that are sub-realms of your realm, but are not actually used. When they are forwarded to you by the proxies, you shouldn’t forward them back to the proxy servers, because that will create loops.

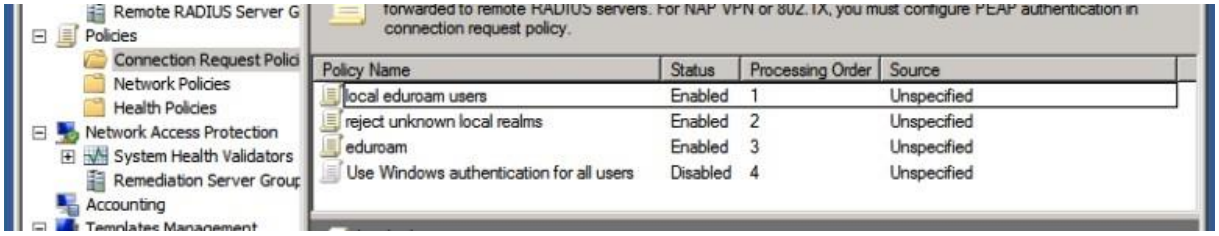


The condition to match for this rule (that should come after all other rules for local users but before the proxy userse) should match a user-name ending on your top-level realm. This is done by a regular expression such as \.lab\.govroam\.nl$



Though we're going to reject the request, set it to authenticate on this server. It needs to be processed locally, and not forwarded.

Don't override any of the settings and finalize the policy; you can create a Network Policy to match the requests as well and assign a Reply-Message to log why the request was rejected, but it's no problem to leave that out.



Make sure the order of Connection Request Policies and the Network Policies is correct and test your configuration.

3.8 Testing

For a RADIUS server set up as an IdP, testing is fairly simple. The national govroam RADIUS servers perform regular dummy authentication requests for monitoring purposes that should be visible (as rejected requests) in the logging. If not, check your routing, NAT-ing and firewall(s) (and optionally the VPN tunnel).

In order to test your SP setup, a couple of tools can be used together with the test-account that your organisation received from the govroam operations department:

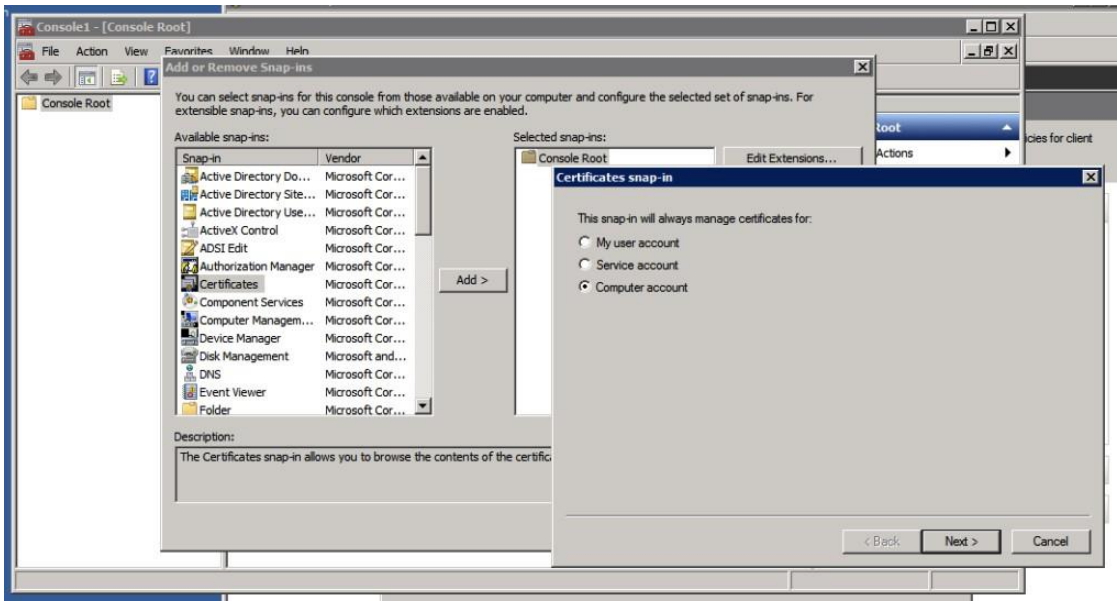
- eapol_test
- radtest

4 Appendix A: Certificates

You need to have a server certificate in order to use PEAP-authentication with govroam. PEAP sets up a secure SSL tunnel (just like HTTPS does for websites) in order to protect the credentials, and is an important part of the mutual authentication: both the user needs to prove who he or she is, and the authentication server needs to prove to the user that he or she is providing credentials to the right authority.

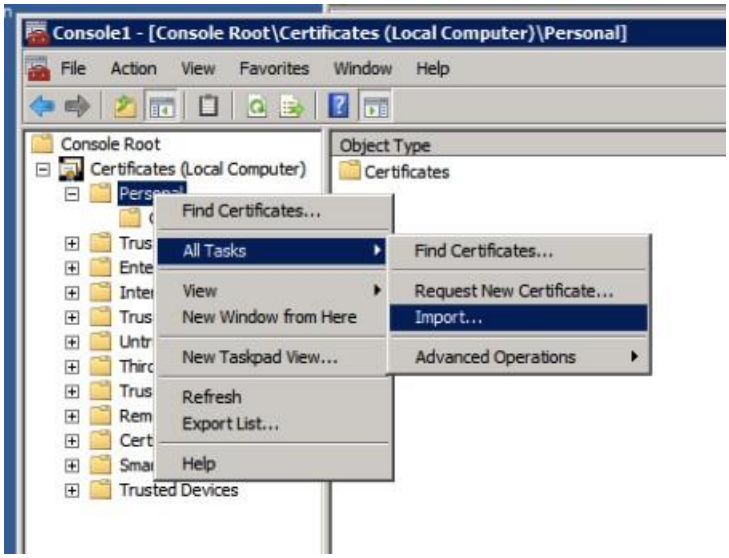
Without certificate (self signed or not) it's not possible to do local authentication. NPS can still be used as a proxy to receive requests from Access Points, log, filter, and forward to the govroam infrastructure.

Open the Microsoft Management Console, mmc (via “Start” – “Run” – “mmc”). Go to “File”, “Add/Remove Snap-in...”, select “Certificates”, click “Add >” and answer the prompt by choosing “Computer account”:

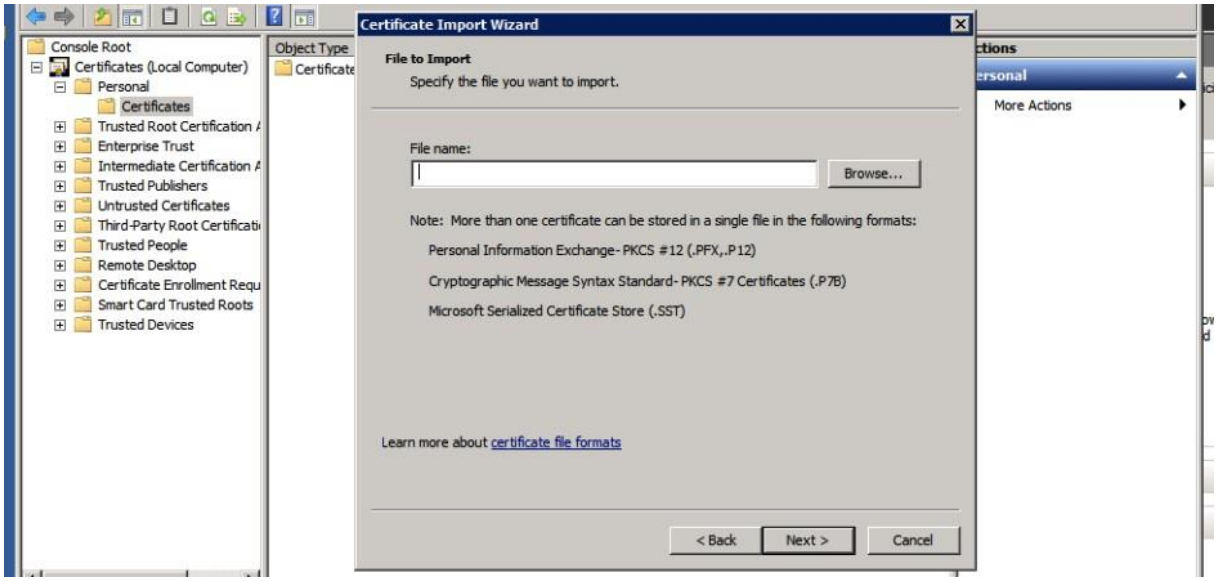


After this, select you want to access the resources on the Local Computer (assuming that's where you install your NPS on), and click “Ok” in the “Add or Remove Snap-ins” window to work with the MMC console.

If you have a signed certificate already in pkcs12 format, you can import it (and/or intermediate certificates) to the “Personal” store by right-clicking the “Personal” folder and choosing “Import...” under “All Tasks”.



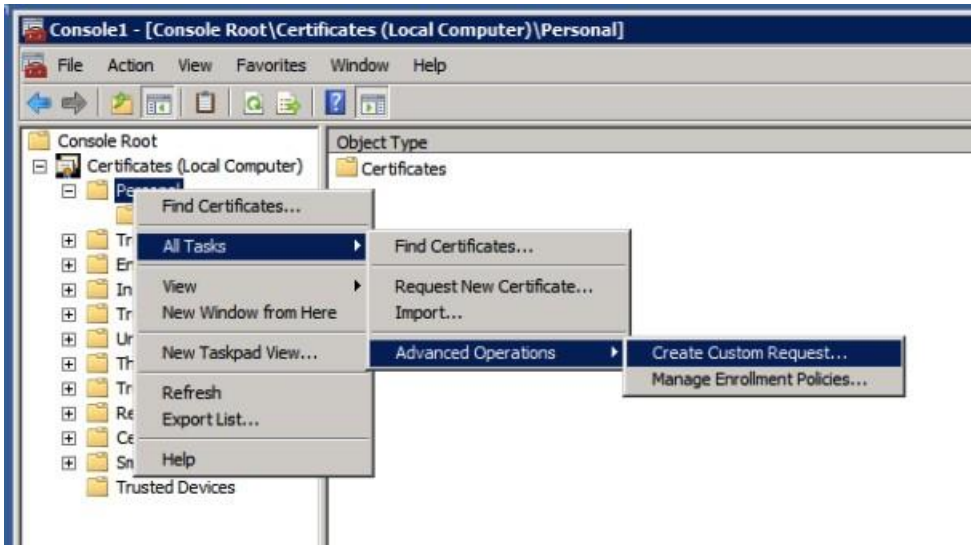
Clicking “Next” after the Certificate Import Wizard introduction asks you for the certificate files to import.



In the next screens you're asked for the password that protected the file, and folder to store the certificate in (this is the “Personal” folder that you just selected). Then the import is complete you will find your certificate in the Personal folder, and you can select it from NPS later.

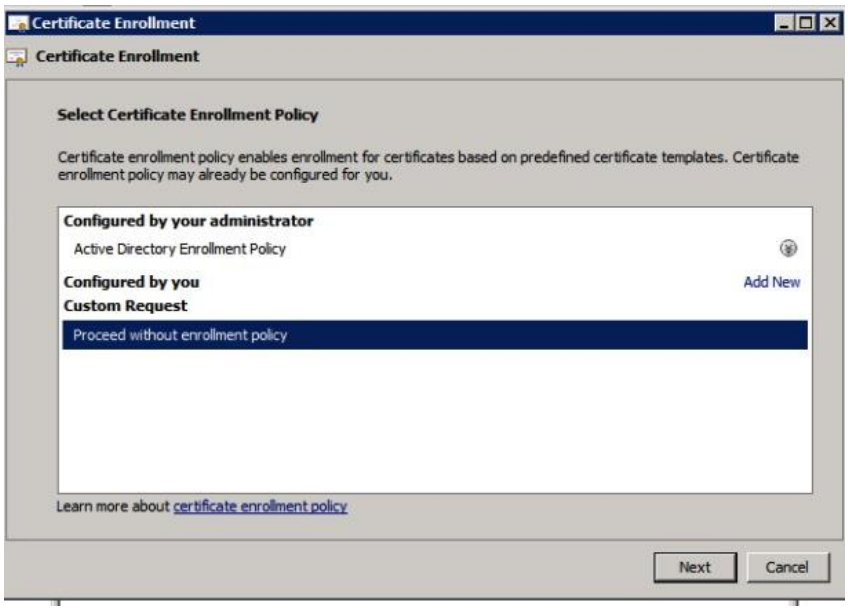
4.1 Generate a certificate request

If you have no existing certificate to import, you need to generate a CSR to be signed.

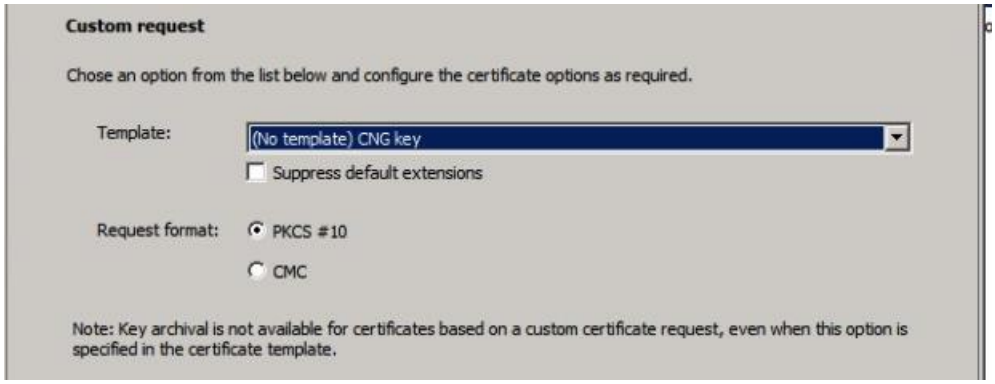


Create the request by right-clicking the “Personal” tree in the Certificates snap-in, selecting “All Tasks – Advanced Operations” and “Create Custom Request”. Click “Next” after the introduction and (assuming you have no internal CA running) choose “Custom Request, proceed without enrollment policy” as shown below.

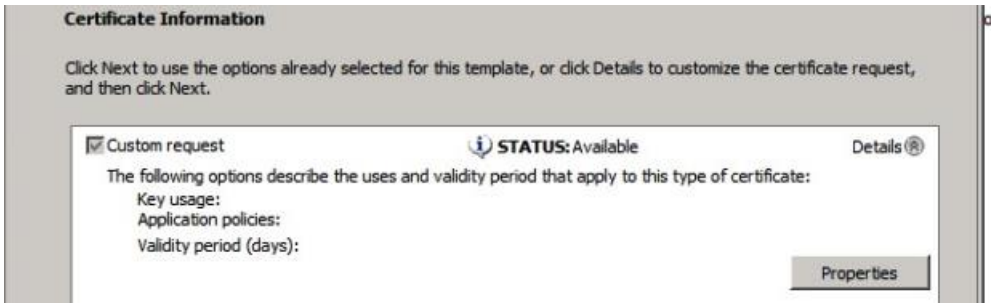
If you have an internal CA, the procedure is different. If your (Windows) clients also get this CA enrolled by the Active Directory, an internal CA might be an option for your server certificate. If your clients (especially true with “bring your own” devices) don't have the internal certificate, having a certificate from a public certificate authority (CA) makes the configuration of govroam on the devices easier. Windows for one, refuses to authenticate if it can't verify the certificate used by any of its stored CA's, whether public or not. A self-signed certificate, means more work for the end-users (and maybe more support calls).



After clicking “Next”, leave the options for the Custom request default to PKCS10;



In the next screen though, you need to change some properties for the requested certificate:



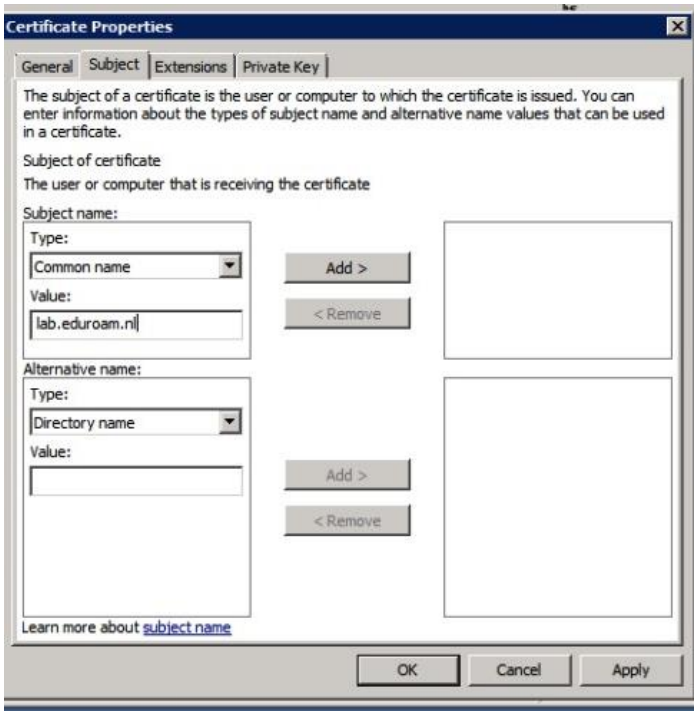
In the “General” tab you can configure a friendly name for the certificate; in this example the “common name” (CN) of the certificate is used: lab.govroam.nl – this is also the domain under which the Active Directory operates, and it will be the RADIUS realm too.

In the “Subject” tab, enter the “Common Name” for your certificate. You probably need to prove ownership of the hostname/domain used to the certificate authority (CA), plus your users will see this name in the certificate: so this name is important.

It doesn't need to be the name of the host itself: actually, if you have multiple NPS servers, it's important that all servers have the same certificate because devices will (at least) prompt when there is a certificate change (which is what then happens during failover).

If your users recognize the name of the certificate when they're prompted, that's probably safer and easier for instructions. (Besides, they might need to check other properties of the certificate, eg. the fingerprint, which is what the Windows 8 client will show for verification.)

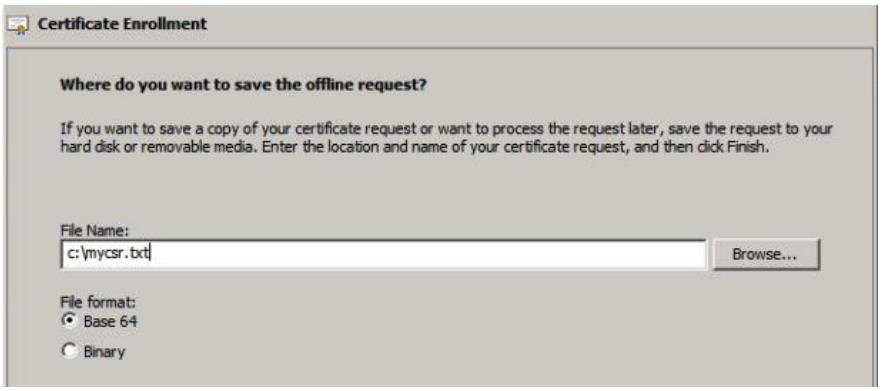
If you make the certificate “govroam.your-org.tld” for instance, that's fine. In this case, we're using “lab.govroam.nl” for the CN.



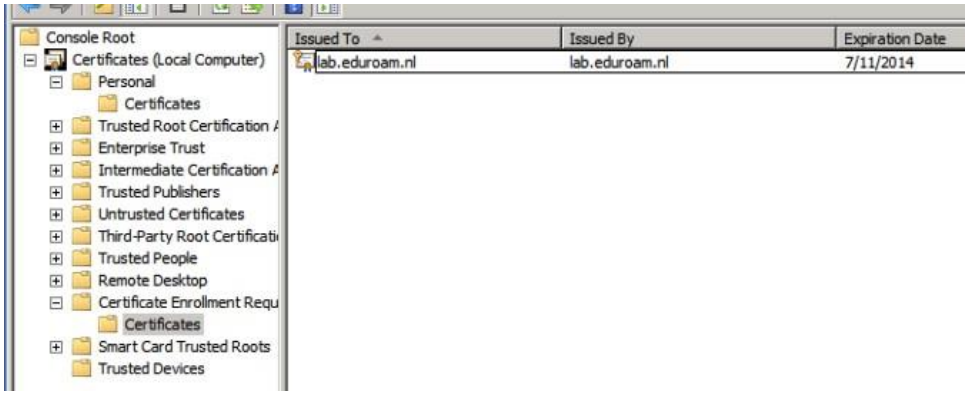
Click “Add” for the subject, and go to the Private Key menu. It's recommended (and by some CA's required) to make the key size 2048 bits. Click “OK” after these changes, and proceed to the “Next” step in the Wizard.

(Make sure the profile used at the public CA includes the TLS server extensions. If you use an internal CA in your Active Directory, you might want to include these extensions in the “Extensions” tab. For a public CA, you probably don't have to worry about this.)

Store your certificate signing request (CSR) in a file: to request the certificate you need to copy-paste the BASE64 contents to the request page.



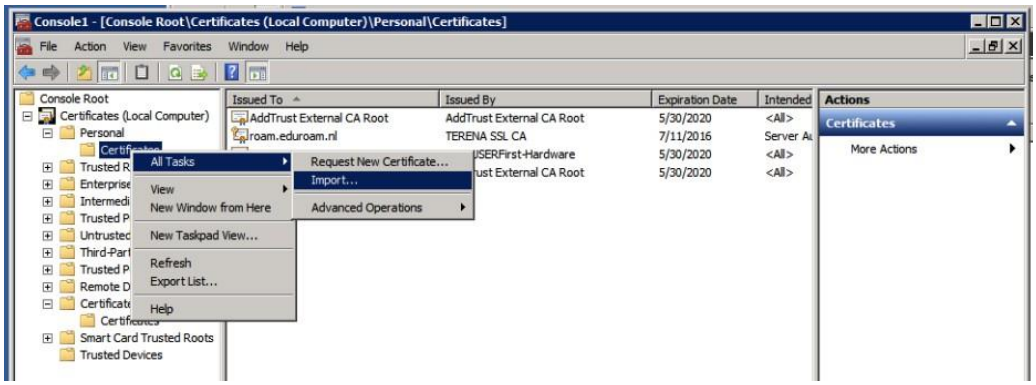
The certificate request (and private key for now), you will find under the Certificate Enrollment Requests. You can also delete it from there if you made a mistake.



Now, request your certificate using the file/BASE64 copy-pasted content at your CA page. If you're a Terena Certificate Service you probably know the URL for this; you can also request a certificate at any of the well-known commercial CA vendors, such as Comodo, GlobalSign, Verisign... or even test it with a trial-certificate that's valid for limited time.

It's preferable to make your certificate expire after a longer period, such as 3 years: your users might receive a prompt about the new certificate that you need to tell them about before changing it. (Normally if they get such a prompt, this could be a man-in-the-middle attack, so inform them about that too! Mutual authentication is an important part of your security!)

After your certificate is issued by the CA, import it via the MMC snap-in:



You can download the .pem files provided by the provisioning interface of the public CA.

Also install the intermediate certificates that you receive from your CA, in particular if they're not already installed in your store. The NPS server needs to send them (along with the certificate) to the clients in order to do proper verification. (This isn't different from protecting a website with SSL certificates.

5 Appendix B: Terms and abbreviations

AAA(A)	Authentication, Authorisation, Accounting (& Auditing)
AD	Microsoft Active Directory
AP	Access Point
CA	Certificate Authority
DHCP	Dynamic Host Configuration Protocol
EAP	Extensible Authentication Protocol
End User	individual or person using the govroam service
IEEE	Institution of Eletrical and Eletronics Engineers
IdP	Identity Provider, the home organisation of the End User
IP	Internet Protocol
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
NAC	Network Access Control, combining 802.1X and RADIUS
Participant	Organisation (not an indivual) participating in the govroam community
RADIUS	Remote Authentication Dial-In User Server/Service/System
SSID	Service Set IDentifier
SP	Service Provider, the organisation providing wifi access
TLD	Top Level Domain
TLS	Transport Layer Security
User	Organisation (not an indivual) participating in the govroam community
VLAN	Virtual LAN
WLAN	Wireless LAN
WPA	Wireless Protected Access