

29/10/08

Deliverable DJ5.1.5,3: Inter-NREN Roaming Infrastructure and Service Support Cookbook - Third Edition



Deliverable DJ5.1.5,3

Contractual Date: 30/04/2008
Actual Date: 29/10/2008
Contract Number: 511082
Instrument type: Integrated Infrastructure Initiative (I3)
Activity: JRA5
Work Item: 1 - Roaming
Nature of Deliverable: R (Report)
Dissemination Level: PU (Public)
Lead Partner: RESTENA
Document Code: GN2-08-230

Authors: S. Winter (RESTENA), T. Kersting (DFN), P. Dekkers (SURFnet), L. Guido (FCCN), S. Papageorgiou (NTUA/GRNET), Janos Mohacsi (NIIF/HUNGARNET), R. Papez (ARNES), M. Milinovic (CARNet/Srce), D. Penezic (CARNet/Srce), J. Rauschenbach (DFN), J. Tomasek (CESNET), K. Wierenga (SURFnet), T. Wolniewicz (Nicolaus Copernicus University, Torun), José-Manuel Macias-Luna (RedIRIS), I. Thomson (DANTE), JRA5 group

Abstract

This Deliverable provides a collection of guidelines and installation instructions ("cookbook") for implementing an eduroam compatible infrastructure.

Table of Contents

0	Executive Summary	1
1	Introduction	2
2	eduroam in a nutshell	3
2.1	General overview	3
2.2	Elements of the eduroam infrastructure	5
2.2.1	Confederation top-level RADIUS Server (TLR)	5
2.2.2	Federation-Level RADIUS servers (FLRs)	5
2.2.3	IdP and SP RADIUS servers	6
2.2.4	Suplicants	6
2.2.5	Access Points	6
2.2.6	Switches	6
3	Example of eduroam Setup	7
3.1	Common FLRS proxy setups	7
3.1.1	Radiator as federation proxy	7
3.1.2	radsecproxy	14
3.1.3	Obtaining certificates for RadSec for federation-level servers	21
3.2	Reference Campus Setup	22
3.2.1	Introduction	22
3.2.2	Configuring the Ethernet switch for eduroam	24
3.2.3	Setting up the RADIUS server	25
3.2.4	RadSec: Using radsecproxy as an add-on to an existing RADIUS server	32
3.2.5	Obtaining RadSec certificates	36
3.2.6	Configuring the Access Point for eduroam	36
3.2.7	Suppliant	40
3.3	Operating the Infrastructure	46
3.3.1	Regular safety precautions	46
3.3.2	Tracking malicious users	46

4	Conclusions	48
5	References	49
6	Acronyms	50
Appendix A	RADIUS servers	52
A.1	Radiator institutional server	53
A.2	FreeRADIUS institutional server	53
A.2.1	Setting up FreeRADIUS	53
A.2.2	Defining clients - Access Points and RADIUS servers in <i>clients.conf</i>	54
A.2.3	Configure realm handling and proxying: <i>proxy.conf</i>	55
A.2.4	Virtual server definition for eduroam: <i>sites-enabled/eduroam</i>	56
A.2.5	Main configuration file: <i>radiusd.conf</i>	57
A.2.6	User authentication: configuring an eduroam IdP	59
A.2.7	Setting up accounting in the SQL database	62
A.2.8	Logging the client IP address as Service Provider (Optional)	63
A.2.9	More information	63
A.3	VitalAAA Institutional and national server	64
A.3.1	Institutional VitalAAA Server (ver. 5.2.0+)	64
A.3.2	National VitalAAA Server (ver. 5.2.0+)	65
A.4	Microsoft Internet Authentication Service server as institutional server	66
A.4.1	Installing IAS	66
A.4.2	Configuring remote RADIUS servers	68
A.4.3	Configuring IAS to act as a university RADIUS server in the eduroam hierarchy	70
A.4.4	Configuring Domain Users to be able to use eduroam with their credentials to Windows Domain	76
A.4.5	Configuration of Authentication methods	77
A.4.6	Troubleshooting	78
A.4.7	References	79
Appendix B	Access Points	80
B.1	Cisco Aironet 1200 Series example setup	80
B.2	LANCOM L-54 Series Access Points	80
B.2.1	NTP setup (confederation requirement: reliable timing source)	81
B.2.2	Logging	82

	B.2.3	Configuring the SSID	82	
	B.2.4	WPA Enterprise security	84	
	B.2.5	RADIUS accounting server (optional)	85	
	B.2.6	Using RadSec instead of RADIUS (optional)	86	
Appendix C		SupPLICants	88	
	C.1	SecureW2	88	
		C.1.1	Installing the CA certificate	88
		C.1.2	Installing SecureW2	90
		C.1.3	Configuring SecureW2	91
		C.1.4	Connecting to eduroam	96
	C.2	MacOS	96	
	C.3	iPhone	101	
		C.3.1	Using the configuration utility	102
		C.3.2	Downloading a profile	105
	C.4	WPA_SupPLICant	107	
	C.5	Intel PROSet/Wireless supplicant	109	
		C.5.1	Preparing the profile as an administrator	109
		C.5.2	Installing the profile as a user	109
Appendix D		eduroam along with commercial hotspot system	110	
	D.1	Installation Instructions	111	

Table of Figures

Figure 2.1: Layers of the eduroam RADIUS hierarchy	5
Figure 3.1: Network Topology	23
Figure A 1: Message flow in RADIUS server Identity Management System	52

0 Executive Summary

This document provides instructions and advice for installing an eduroam implementation. Readers are assumed to be either system-administrators or advanced users, and therefore to have an in-depth knowledge of the technology and software involved.

There are many options available when building an eduroam infrastructure (especially at the campus level). Therefore, this deliverable is presented in the form of a “cookbook”; it provides first a general recipe for an implementation (sections 1 to 4), followed by a range of implementation examples to help configure an installation to specific requirements (the Appendices).

To help configuration, a description of a reference campus is provided in section 3.2 “Reference Campus Setup”. This is based on products frequently used in the JRA5 research and education area, ensuring a high level of practical assurance concerning the instructions given. However, it is important to note that this example does not provide a definitive list of which technical equipment to buy or to use, nor does it imply any recommendations. Other solutions might well provide the same level of functionality, or even higher.

Finally, please note that this deliverable is not meant to describe eduroam and the underlying architecture in detail, as these descriptions can be found in the deliverable DJ5.1.4 (“Inter-NREN Roaming Architecture: Description and Development Items”).

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

1 Introduction

This document gives detailed information and advice on how to implement the eduroam architecture, described in GN2-06-137 DJ5.1.4 "Inter-NREN Roaming Architecture: Description and Development Items" [GN2DJ514], in a real-world environment. It is aimed at system administrators who need to implement eduroam at their institution. As such an in-depth knowledge of roaming technologies and equipment is assumed.

The purpose of the "cookbook" format adopted in this document is not to promote any product, but to select popular solutions with a broader usage rate in the research and education area as examples to describe the installation procedures. The idea is to enable the campus administrator or the eduroam operator to adapt these configuration principles for the devices operated in their home environment.

The "cookbook" format consists of:

- Chapter 2: An overview of the necessary, basic elements needed to implement an eduroam network, as well as their specific functions.
- Chapter 3: The configuration of a typical example setup, ranging from national RADIUS servers down to user supplicants.
- Appendices A to C: Configuration examples for multiple variations of the elements described, covering RADIUS servers, switches, access points and supplicants of different vendors or available as open source solutions.

The list of products for which guidelines are provided is not exhaustive, and might also depend on the version of the software used.

Changes in the context of technical progress are unavoidable, therefore please be aware that this deliverable has an inherent dynamic aspect and will be amended as new technologies and advancements are implemented.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

2 eduroam in a nutshell

2.1 General overview

Please refer to deliverable DJ5.1.4 "Inter-NREN Roaming Architecture: Description and Development Items" for an in-depth description of eduroam and the underlying architecture.

Eduroam stands for EDUcation ROAMing. It offers users from participating academic institutions secure Internet access at any other eduroam-enabled institution. The eduroam architecture that makes this possible is based on a number of technologies and agreements, which together provide the eduroam user experience: "open your laptop and be online".

The crucial agreement underpinning the foundation of eduroam 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. Earlier versions of this document used the term "home institution" to refer to the Identity Provider, but this term is considered deprecated.
- The authorisation decision allowing access to the network resources upon proper authentication is done by the **owner** of the **visited network**, also called the **Service Provider (SP)**. Earlier versions of this document also used the deprecated term "visited institution".

In order to transport the authentication request of a user from the Service Provider to his Identity Provider and the authentication response back, a hierarchical 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 central national RADIUS server, which in turn is connected to a European (or global) RADIUS server. 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=country code top-level domain), the RADIUS servers can use this information to route the request to the appropriate next hop in the hierarchy until the IdP is reached. An example of the RADIUS hierarchy is shown in Figure 2.1.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

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). Using the appropriate EAP-method either a secure tunnel will be established from the user's computer to their IdP through which the actual authentication information (username/password etc.) will be carried (EAP-TTLS or PEAP), or mutual authentication by public X.509 certificates (which is not vulnerable to eavesdropping) will be used (EAP-TLS).

RADIUS transports the user's name in an attribute *User-Name*, which is visible in cleartext. It also transports the EAP payload, which is encrypted and not visible to intermediate servers, only to the IdP server. In order to ensure privacy, it might be desirable not to put the real username in the RADIUS User-Name attribute (this attribute is the "outer" identity). Instead, it might be preferred to put *@realm* in this attribute (nothing left of the *@* - this is the IETF-suggested format). The realm part still must be the correct one as it is used to route the request to the respective home server. Once the IdP server decrypts the TLS tunnel in the EAP payload, it gets the real user name - the "inner" identity.

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.

Note: On responsibility for actions of the user: Directive 2001/31/EC article 12 defines the liability of a service provider:

1. Where an information society service is provided that consists of the transmission in a communication network of information provided by a recipient of the service, or the provision of access to a communication network, Member States shall ensure that the service provider is not liable for the information transmitted, on condition that the provider:
 - (a) does not initiate the transmission;
 - (b) does not select the receiver of the transmission; and
 - (c) does not select or modify the information contained in the transmission.
2. The acts of transmission and of provision of access referred to in paragraph 1 include the automatic, intermediate and transient storage of the information transmitted in so far as this takes place for the sole purpose of carrying out the transmission in the communication network, and provided that the information is not stored for any period longer than is reasonably necessary for the transmission.
3. This Article shall not affect the possibility for a court or administrative authority, in accordance with Member States' legal systems, of requiring the service provider to terminate or prevent an infringement.

The complete Directive can be found at EUR-Lex¹.

¹ <http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32000L0031:EN:HTML>

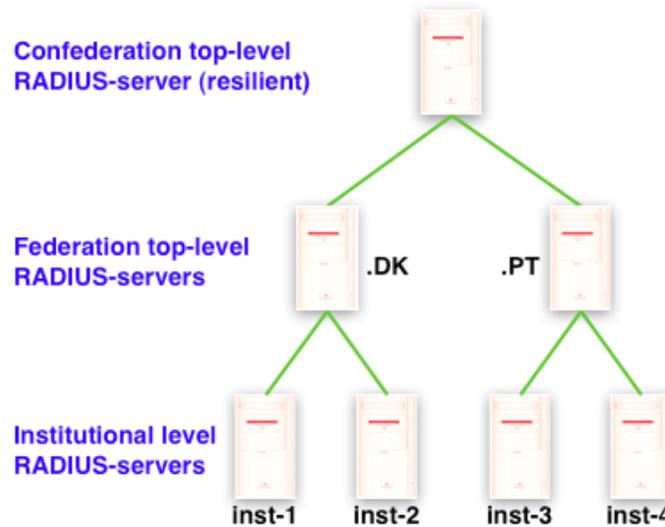


Figure 2.1: Layers of the eduroam RADIUS hierarchy

2.2 Elements of the eduroam infrastructure

2.2.1 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.

2.2.2 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.

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

2.2.3 IdP and SP RADIUS servers

The IdP RADIUS server is responsible for authenticating its own users (at its own premises, if it also an SP, or when they are visiting another SP) by checking the credentials against a local identity management system.

The SP RADIUS server is 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.

Note that the IdP RADIUS server is the most complex of all. 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.

The Identity Management System contains the information of the end users; for instance usernames and passwords. They must be kept up-to-date by the responsible IdP.

2.2.4 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).

2.2.5 Access Points

Access Points are Wireless LAN access devices conformant to IEEE 802.11 and need to be IEEE 802.1X capable. 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.

2.2.6 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.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

3 Example of eduroam Setup

The following sections provide a sample setup of the eduroam network from top to bottom. To begin with, the configuration for a Federation-Level RADIUS Server (FLRS) is presented, followed by the various components needed at an SP and IdP level, right down to the supplicants of the end-users.

3.1 Common FLRS proxy setups

This section covers the configuration of a Federation RADIUS Proxy server. It presents two different software packages to achieve that goal: the Radiator (<http://www.open.com.au/radiator/>) RADIUS server and the free software package radsecproxy (<http://software.uninett.no/radsecproxy/>).

Examples in this chapter can be used as a repository of configuration snippets for building complex proxy servers, or can be used as described for simple proxy relations between one organisation and two top-level servers.

3.1.1 Radiator as federation proxy

3.1.1.1 Common configuration

Radiator expects the configuration to be in file `/etc/radiator/radius.cfg`

```
LogDir          /var/log/radiator
DbDir           /usr/share/radiator
```

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

LogDir defines the directory in which start-up logs and PID file reside. DbDir defines the path to Radiator's data files, such as dictionaries.

Trace 3 logs will be sent to the system syslog, and Trace 4 logs will be stored in the directory /var/log/arch/radiator/

Here the location of logfiles (referred in the configuration as %L) and databases (%D) can be defined, as well as the amount of log output (with Trace).

The log files will be split on a daily basis.

```
<Log SYSLOG>
    Facility    local7
    LogIdent    log-syslog
    Trace       3
</Log>
<Log FILE>
    Filename    /var/log/arch/radiator/radiator.%Y_%m_%d.log
    LogIdent    log-file
    Trace       4
</Log>
```

SNMP allows remote monitoring of activity on a RADIUS server with tools such as RADAR from OSC (<http://www.open.com.au/radar/index.html>), or drawing simple graphs of activity by rgraph from CESNET (<http://www.eduroam.cz/rgraph/>).

```
<SNMPAgent>
    ROCommunity  xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
    Managers     localhost 127.0.0.1
</SNMPAgent>
```

The ports Radiator will use to listen for Authentication and Accounting requests must be defined. The port numbers 1812 and 1813 were assigned to the RADIUS protocol by IANA. Unfortunately, some RADIUS servers (such as CiscoACS 3.x) are still using the old numbers 1645, 1646. For this reason it is suggested to use both numbers.

```
AuthPort      1645,1812
AcctPort      1646,1813
```

It is possible to define own logging formats. We suggest using the following log definition – it generates one single line of log output per authentication, which is very parser-friendly if logs need to be evaluated later:

```
<AuthLog SYSLOG>
```

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
Identifier          defaultAuthLog

Facility            local7
LogIdent            radiator
SuccessFormat       access-accept for %u (User-Name=%{Reply:User-Name}) at
Proxy=%c (CSID=%{Calling-Station-Id} NAS=%{NAS-Identifier}/%N)
EAP=%{HexAddress:EAP-Message}
FailureFormat       access-reject for %u (User-Name=%{Reply:User-Name}) at
Proxy=%c (CSID=%{Calling-Station-Id} NAS=%{NAS-Identifier}/%N)
LogSuccess           1
LogFailure           1
</AuthLog>
```

3.1.1.2 Client definition

In the client section all possible peers (the “institutional” and the confederation RADIUS servers) have to be listed and a “secret” has to be assigned to them. As this secret is the only thing that protects the communication between the RADIUS servers from eavesdropping, it must be cryptographically strong and well protected.

```
<Client localhost>
  Secret      mysecret
  DupInterval 0
</Client>

<Client radius.orgA.tld>
  Secret      XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  Identifier   radius.orgA.tld
</Client>

<Client etlr1.eduroam.org>
  Secret      XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  Identifier   etlr1.eduroam.org
</Client>

<Client etlr2.eduroam.org>
  Secret      XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  Identifier   etlr2.eduroam.org
</Client>
```

It is necessary to mark each host by a unique identifier, which is later used to prevent loops in the hierarchy (note that this identifier is for local use only, therefore select an identifier that is meaningful to your

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

implementation).

etlr1.eduroam.org and etlr2.eduroam.org are the European top-level RADIUS servers.

3.1.1.3 Handling empty realms

Theoretically, empty realms should not reach a National RADIUS Proxy, but if they do this code will prevent the National RADIUS Proxy from sending these empty realms to the international top-level proxy servers. A reply will be added to the rejected requests explaining the reason for rejection. Replace <TLD> with the federation top-level domain you are authoritative for.

```
<Handler Realm=/^$/>
    AccountingHandled
    StripFromReply Reply-Message
    AddToReply Reply-Message="Misconfigured client: empty realm! Rejected
by <TLD>."
    AuthLog defaultAuthLog
</Handler>
```

3.1.1.4 Proxying to an organisation with one RADIUS Server

In the case in which an organisation has only one server, configuration is relatively easy:

```
<Handler Realm=/^orgA\.tld$/i,
    Client-Identifier=/^(?!radius.orgA.tld$)/>
    <AuthBy RADIUS>
        RetryTimeout          3
        Retries                1
        FailureBackoffTime    0

        UseExtendedIds

        <Host radius.orgA.tld>
            AuthPort          1812
            AcctPort          1813
            Secret             xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
        </Host>
    </AuthBy>
```

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
AuthLog defaultAuthLog
```

```
</Handler>
```

Realms should be pattern matched case-insensitive, because users sometimes type their realms using upper case letters.

The configuration directive “Client-Identifier=/(?!radius.orgA.tld\$)” blocks RADIUS packets from radius.orgA.tld with realm orgA.tld from entering this Handler.

If such a packet is received it will be rejected by the handler matching realm=/.*\tld\$/i. This prevents loops between the national proxy and the server of orgA.tld.

Servers to proxy should never be marked as dead, because in doing so Radiator will not try to communicate with them until `BackoffTime` expires, even if that server is already live again.

3.1.1.5 Proxying to multiple RADIUS servers

Top-level RADIUS servers (confederation or federation) are duplicated for better reliability of the eduroam service, as required in the policy document [GN2DJ513,2]. Also some organisations have multiple RADIUS servers for the same reason. To use multiple servers in eduroam, a special setup has to be used. Standard detection of “dead” hosts based on timeout will not work here; timeouts might be caused by the infrastructure behind the direct peer of RADIUS server, so the server cannot assume that its peer is off-line. But if it really is down, a switch to the backup server must be performed. Unfortunately the RADIUS protocol does not provide a clear way to do that.

The problem can be solved by using a special configuration developed at CESNET and described in the article “Dead-realm marking feature for Radiator RADIUS servers” (<http://www.eduroam.cz/dead-realm/docs/dead-realm.html>).

The idea behind the dead-realm marking is quite simple. Instead of marking the server dead for all realms (as in dead-host marking) just the particular realm on the host is marked dead and requests are routed to another configured host.

The variable `dr_timeout` controls how long a realm will be marked dead on a particular host. 3600 seconds is the suggested value. Much lower values will cause the system to check the dead primary server too often. Higher values can be considered if the backup server is equally as powerful as the primary server.

```
StartupHook sub { require "/etc/radiator/DeadRealm.pm"; }  
DefineFormattedGlobalVar      dr_timeout 3600
```

Following the two `AuthBy` sections are definitions of the servers, which will be used as a pool for destinations defined later.

`NoReplyHook` and `ReplyHook` are necessary for the dead-realm marking feature, a Perl implementation is

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

available online at <http://www.eduroam.cz/dead-realm/>. A copy version 2.0 of this implementation is in the ZIP file which accompanies this document.

NoReplyHook is called when no reply is received from the RADIUS server after two attempts within six seconds. In this case the processed packet is ignored, the realm is marked as dead on this server and when a subsequent packet with the same realm is received it will be forwarded to the other defined server.

ReplyHook is called when a response from the server is received; if the realm is marked as dead it will be unmarked and thereby identified as live.

```
<AuthBy RADIUS>
  Identifier                etlr1.eduroam.org

  RetryTimeout              3
  Retries                   1
  FailureBackoffTime       0

  UseExtendedIds

  <Host etlr1.eduroam.org>
    AuthPort                1812
    AcctPort                1813
    Secret                  XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  </Host>

  NoReplyHook sub { DeadRealm::noReplyHook(@_); };
  ReplyHook sub { DeadRealm::replyHook(@_); };
</AuthBy>
```

```
<AuthBy RADIUS>
  Identifier                etlr2.eduroam.org

  RetryTimeout              3
  Retries                   1
  FailureBackoffTime       0

  UseExtendedIds
```

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
<Host etlr2.eduroam.org>
    AuthPort      1812
    AcctPort      1813
    Secret        xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
</Host>

    NoReplyHook sub { DeadRealm::noReplyHook(@_); };
    ReplyHook sub { DeadRealm::replyHook(@_); };
</AuthBy>
```

3.1.1.6 Handling unknown realms in the own federation

After the last connected realm, it makes sense to reject all known-bad realms and all unknown realms which end in the own federation's top-level domain. One such invalid realm is seen quite often due to supplicant misconfiguration: myabc.com (this is the default realm in an unconfigured Intel PRO/Set Wireless supplicant). The following stanza rejects this realm with an appropriate error message and blindly acknowledges all Accounting requests.

```
<Handler Realm=/myabc\.com$/i>
    AccountingHandled

    StripFromReply Reply-Message
    AddToReply      Reply-Message=" Misconfigured client: default realm of
Intel PRO/Wireless supplicant! Rejected by <TLD>."
    AuthLog defaultAuthLog
</Handler>
```

Add the following stanza after the last valid realm to catch and reject all unknown realms that end in the own TLD:

```
<Handler Realm=/.*\.tld$/i>
    AccountingHandled

    StripFromReply Reply-Message
    AddToReply      Reply-Message=" Misconfigured supplicant or downstream
server: uses non-existing realm in <TLD> federation!"
    AuthLog defaultAuthLog
</Handler>
```

The following directive (DefineFormattedGlobalVar) defines the variable dr_TOPLEVEL_server_list and assigns

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

a list of servers responsible for the handler marked as TOPLEVEL to this variable.

```
DefineFormattedGlobalVar dr_TOPLEVEL_server_list \  
    etlr1.eduroam.org,etlr2.eduroam.org
```

The handler with the filter `Realm=/^.+$/` will receive any request that will not be caught by more specific filters before (`Realm=/^orgA.tld$/i`, `Realm=/.*\.tld$/`, `Realm=/^$/`). In this example it will be all requests that belong to other TLD's than the ones defined on this server – requests that have to be forwarded to top-level servers.

RequestHook will use the value of `Identifier (= TOPLEVEL)` defined in this handler to search the variable `dr_TOPLEVEL_server_list` holding the list of servers. It will forward the request to a server not marked dead, or if all are marked dead the one with the oldest dead-realm mark.

```
<Handler Realm=/^.+$/>  
    Client-Identifier=/^(?!etlr1.eduroam.org$)/>  
    Client-Identifier=/^(?!etlr2.eduroam.org$)/>  
  
    Identifier TOPLEVEL  
  
    <AuthBy INTERNAL>  
        RequestHook sub { DeadRealm::chooseServer(@_); }  
    </AuthBy>  
    AuthLog defaultAuthLog  
</Handler>
```

3.1.2 radsecproxy

3.1.2.1 Goals and Prerequisites

This section describes how to set up radsecproxy to act as a federation-level RADIUS+RadSec server. It can then completely replace other RADIUS server products on the federation level.

More precisely, it will enable a server to:

- Accept requests from connected service providers via RADIUS and RadSec.
- Forward requests to connected identity providers via RADIUS and RadSec.
- Forward requests from international visitors to the European eduroam confederation root servers via RadSec.
- Accept requests from the root servers via RadSec for the own federation's users when they are roaming

in another federation.

The prerequisites for this deployment are:

- radsecproxy version 1.2 or higher
- A server certificate and a private key for that certificate to establish the RadSec connection which designates the server as a federation proxy:

```
urn:geant:eduroam:component:proxy:Europe:<FedName>:<id>
```

3.1.2.2 Sample configuration

Most of the radsecproxy configuration file is static. Therefore, a template configuration file is provided at <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>. A detailed explanation of this configuration file follows. However, the comments included in the file should make its action almost self-explanatory. This means you can start and experiment with it right after installation.

3.1.2.3 Installation

On UNIX-like systems, the installation is very simple.

1. Download the code from <http://software.uninett.no/radsecproxy/>.
2. Unpack the code.
3. Navigate into the unpacked directory (the base directory) and type:
`make`

The code is ISO C and should compile cleanly. It usually does not require a `./configure`.

4. After compiling, the executable:radsecproxy is in the base directory.
Either run this executable here or copy it to a convenient location (e.g. `/usr/local/bin`) and run it there.
Execution does not require root rights. It expects its configuration file to be in:

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

5. Create the directory `/etc/radsecproxy.d/certs/ca/`. The template configuration file requires this directory: to contain the accredited CA root certificates and the corresponding Certificate Revocation Lists (CAs) in their OpenSSL hash form.
6. Currently, two CAs are accredited for eduroam. These two certificates and their CRLs need to be copied into this directory. For your convenience, we have prepared a script at <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>, which you can execute in the directory. It will download and hash the appropriate files. In this case you can skip steps 7 and 8.
7. If you want to download the files manually, download them from these URLs:
 - <http://sca.edugain.org/cacert/eduGAINCA.pem>

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

- <http://sca.edugain.org/cacert/eduGAINSCA.pem>
 - <http://pki.edugain.org/edugainca/crl/cacrl.pem>
 - <https://sca.edugain.org/crl/cacrl.pem>
8. After downloading, execute the following command in the `/etc/radsecproxy.d/certs/ca/` directory:
`c_rehash .`
(including the dot) to generate OpenSSL hashes of the downloaded files.
 9. Make sure that the CRL files are regularly re-downloaded. This can for example be done by executing the download script regularly with cron. CRLs have a limited validity time themselves – not regularly refreshing the CRLs will eventually cause certificate validation to fail!
 10. Copy the template configuration file into `/etc/radsecproxy.conf`.
 11. Fill the lines marked with `__STUFF__`.
 12. Start radsecproxy and enjoy (for first-time use, starting it with the `-f` option is recommended, it will start radsecproxy in the foreground and show some verbose startup messages).

3.1.2.4 Detailed explanation of configuration

This walkthrough goes through the template `radsecproxy.conf` line by line and explains the meaning of each stanza.

```
ListenUDP          *:1812
ListenTCP          *:2083
```

radsecproxy will receive requests from all connected Service Providers via both RADIUS and RadSec. Therefore it has to listen on the appropriate ports on its network interfaces (the * meaning: all interfaces). If you want radsecproxy to listen only on specific interfaces, enter the interface names here. Beware: in this case you may also have to set the more exotic options `SourceUDP` and/or `SourceTCP` (see the man page of radsecproxy for details).

```
LogLevel           3
LogDestination     file:///var/log/radsecproxy.log
```

A logging level of 3 is the default and recommended log level. Radsecproxy will then log successful and failed authentications on one line each. The log destination is a typical POSIX-compliant log location.

```
LoopPrevention     On
```

This enables a semi-automatic prevention of routing loops for RADIUS connections. If you connect a client {...] and a server {...} and give them the same descriptive name, the proxy will prevent proxying.

```
tls default {
    CACertificatePath      /etc/radsecproxy/certs/ca/
    CertificateFile        /etc/radsecproxy/certs/__CERT_PEM__
    CertificateKeyFile     /etc/radsecproxy/certs/__CERT_KEY__
    CertificateKeyPassword __CERT_PASS__
}
```

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
CRLCheck          On
}

```

This section defines which TLS certificates should be used by default. This installation of radsecproxy always uses the same certificates, so this is the only TLS section. CACertificatePath contains the eduroam-accredited CA certificates with filenames in the OpenSSL hash form. The parameters below need to be adapted to point to your server certificate in PEM format, the private key for this certificate and the password for this private key if needed, respectively. If no password is needed for the private key, you can comment this line (precede it with a # sign). The option CRLCheck validates certificates against the Certificate Revocation List (CRL) of the CAs. It requires a valid CRL in place, or else the certificate validation will fail. Therefore, it is important to regularly update the CRLs by re-downloading them as described above.

```
rewrite defaultclient {
    removeAttribute    64
    removeAttribute    65
    removeAttribute    81
}

```

This section deletes attributes from RADIUS requests that convey VLAN assignment information. If VLAN information is sent inadvertently, it can cause a degraded or non-existent service for the end user because he might be put into the wrong VLAN. Connected service providers should filter this attribute on their own. Connected Identity Providers should not send this attribute at all. Checking for the existence of these attributes on the federation server is just an optional additional safety layer. If you do have a roaming use for cross-institution VLAN assignment, you may want to delete this stanza.

```
client 127.0.0.1 {
    type    udp
    secret  testing123
}

```

```
client ::1 {
    type udp
    secret  testing123
}

```

There is no other RADIUS server running on localhost, which makes these client definitions almost superfluous. They may be of some use however to initiate debugging requests and tests from the server itself, so it is considered good practice to list localhost as a client. If your system is not IPv6-enabled, simply delete the stanza about **client ::1**.

```
client __SP_IP_ADDR__ {
    type UDP
    secret  __SP_SECRET__
}

```

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

Stanzas like this one are used for each connected service provider that is connected via RADIUS. You need to know the IP address of every SP's RADIUS server and negotiate a shared secret with the SP.

```
client incoming {
    host *
    type    TLS
    secret mysecret
    matchcertificateattribute
    SubjectAltName:URI:/^https://registry.edugain.org/resolver/?urn=urn:geant:eduroam:
    component:(sp|proxy|confederation-root).*$/
}
```

All incoming RadSec connections can be handled with this stanza. It does not need to be modified. The eduroam trust model requires that a SP that tries to connect has:

- A X.509 certificate from an eduroam-accredited CA (configured above in block **tls default**).
- A URN proving its eligibility (this is achieved with the `matchcertificateattribute` option).

The traditional RADIUS **shared secret** has no meaning in RadSec and can be statically set without security implications. In current eduroam deployments, it is set to **mysecret**. This may change in the future.

Please note that the client and server stanza for the Service Activity 5 (SA5) monitoring have the same host address, but different stanza names. This is important: it disables the LoopDetection for this host, and the SA5 monitoring deliberate uses loops to do its tests.

```
client SA5-monitoring-incoming {
    host 161.53.2.204
    type UDP
    secret __MONITORING_SECRET__
}
```

This is the eduroam Service Activity's monitoring client. Negotiate a shared secret for monitoring with the operators in SA5 and enter it here.

```
server __DESCRIPTIVE_NAME__ {
    host __IP_ADDR__
    type UDP
    secret __SERVER_SECRET__
}
```

To deliver requests to your connected IdPs, their servers need to be configured. This stanza is for IdP servers using RADIUS.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
server __RADSEC_PEER_DNS_NAME__ {
    type TLS
    secret mysecret
    statusserver on
    matchcertificateattribute
SubjectAltName:URI:/^https://registry.edugain.org/resolver\?urn=urn:geant:eduroam:
component:(idp|proxy|confederation-root):.*$/i
}
```

This is the equivalent stanza for IdP servers using RadSec.

```
server etlrl.eduroam.org {
    type TLS
    secret mysecret
    statusserver on
    matchcertificateattribute \
SubjectAltName:URI:/^https://registry.edugain.org/resolver\?urn=urn:geant:eduroam:comp
onent:confederation-root:Europe:ETLR:.*$/i
}
```

```
server etlr2.eduroam.org {
    type TLS
    secret mysecret
    statusserver on
    matchcertificateattribute \
SubjectAltName:URI:/^https://registry.edugain.org/resolver\?urn=urn:geant:eduroam:comp
onent:confederation-root:Europe:ETLR:.*$/i
}
```

These are the European eduroam Confederation root servers. This entry can be kept as it stands and doesn't need any further configuration.

```
server SA5-monitoring-outgoing {
    host 161.53.2.204
    type UDP
    secret __MONITORING_SECRET__
}
```

```
server TODO:add SRCE RadSec server here {
    type TLS
    secret mysecret
    statusserver on
}
```

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
matchcertificateattribute \  
SubjectAltName:URI:/^urn:geant:eduroam:component:confederation-root:Europe:Monitoring:./i  
}
```

SA5 monitoring works both ways. The client entry near the beginning of the configuration file was needed for incoming requests from the monitoring servers. The two entries here specify the outgoing connections **to** the monitoring servers. Outgoing connections can use either RADIUS (first entry) or RadSec (second entry) at your discretion. You can configure both, and select the preferred way later in the special monitoring realm **eduroam.TLD**.

Finally, the IdP realms need to be defined. This is done in the remainder of the configuration file:

```
realm /myabc\.com$ {  
    replymessage "Misconfigured client: default realm of Intel PRO/Wireless  
supplicant! Rejected by <TLD>."  
}  
  
realm /^[^@]*$ {  
    replymessage "Misconfigured client: empty realm! Rejected by <TLD>."  
}
```

There are some known client-side misconfigurations. If they were not already caught by the SP local RADIUS server, it does not make sense for the proxy to send these requests up to the eduroam infrastructure. These requests are immediately rejected.

Note: if you need to blacklist an existing realm for some reason, you can follow the myabc.com example, copying and replacing it with the realm to be blacklisted.

```
realm /__IDP_REALM__$ {  
    server __FROM_SERVER_STANZAS_ABOVE__  
    server __BACKUP_NAME__  
}
```

Requests that are coming in from upstream and are supposed to be handled by an identity provider are listed in stanzas like this. `__IDP_REALM__` contains the realm of the connected IdP. Create one such stanza for each IdP realm. If an IdP has multiple servers for a failover configuration, you can list all servers in a row, as in the example above.

```
realm /eduroam\.__YOUR_TLD__ {  
    server SA5-monitoring-outgoing  
    server TODO:SRCE RadSec server  
}
```

The configuration stanza above is for outgoing SA5 monitoring connections. You can select your preference for RADIUS or RadSec for the outgoing connections by changing the order of the server stanzas.

```
realm /\._YOUR_TLD__$ {  
    replymessage "Misconfigured supplicant or downstream server: uses known-  
bad realm in <TLD> federation!"  
}
```

All the valid realms were listed earlier in the configuration file, and this server is authoritative for the own TLD. If a supplicant or downstream servers sends a realm with the own TLD, but also with a realm name that is not registered, this request is unauthorised and bound to fail. It will be rejected immediately to prevent routing loops.

```
realm * {  
    server etlr1.eduroam.org  
    server etlr2.eduroam.org  
}
```

Finally, all realms that do not belong to the own federation are forwarded to the European eduroam Confederation root servers.

3.1.3 Obtaining certificates for RadSec for federation-level servers

To be able to use RadSec in eduroam, you need certificates from a trusted CA with the correct URN values. The procedure below describes how to become a Registration Authority (RA) for the eduGAIN-SCA (eduGAIN Subordinated Certification Authority) with the permission to issue eduroam RadSec certificates. Please note that in the course of RadSec deployment, other CAs may also become accredited to issue such certificates, and that these other CAs have their own operational procedures. When in doubt, ask on the SA5 mailing list whether there is a more specialised CA in your own NREN.

First, you as an NRO operator need to register the three eduroam URN branches on the URN registry at <https://registry.edugain.org/eduroam/>

Click on “Browser” on the top navigation bar and browse to

```
urn:geant:eduroam:component:proxy:Europe
```

You need to register your own NRO’s name directly beneath this URN node. You can do this by clicking on the “+” sign to the right-hand side of that node. If you are going to operate your own sub-registry for your own branch, choose “Request a delegation” in the form which appears. If you do not want to do that, choose “Register a URN” instead (this is recommended for small initial deployments). Fill out the form and send it by clicking “Accept”.

If you plan to also issue certificates to IdP and SP servers within your federation, repeat the same registration process also for the URN branches

```
urn:geant:eduroam:component:idp:Europe
```

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
urn:geant:eduroam:component:sp:Europe
```

For a more detailed explanation of the URN values, please refer to chapter 3 of deliverable DS5.4.1: Report on RadSec Integration.

After completing all the registry branch registrations, please wait until you get an e-mail notification that the URN branches have indeed been registered. This may take a few days since it is a manual verification process involving both the registry operators and the eduroam Operations Team.

The next step is to add the newly acquired URNs to your Registration Authority. If your NREN already is an RA, a simple e-mail to the eduGAIN-SCA operators to add the new URNs to the existing RA is enough. If your NREN is not an RA yet, you need to fill out the RA application form first. Please get in touch with the SA5 mailing list if you need to do this step.

You can find out whether your NREN is already an accredited RA by visiting <http://sca.edugain.org> and looking for your NREN in the list of accredited RAs.

After your RA is accredited for issuing certificates in the eduroam branch, requesting a new certificate is simple: visit <http://sca.edugain.org>, click on your RA and select "Certificate request" in the navigation bar on the left-hand side.

3.2 Reference Campus Setup

3.2.1 Introduction

Campus networks vary widely in such things as topology, equipment used, software, and so on. In order to assist a campus administrator in setting up eduroam on their campus, this section presents the implementation of a typical setup. It is hoped that this will allow users of different topologies and/or equipment to understand the necessary steps to take. Furthermore, in the appendices the same setup will be expanded for a number of other common types of equipment and software. Lastly, we are planning to provide these and future example configurations on the website <http://www.eduroam.org>.

For the reference network we use a typical set of network equipment consisting of:

- A Cisco Catalyst 3550 (or similar) switch.
- A Cisco Aironet AP-1200 Access Point.
- A laptop with Windows XP.
- A Radiator RADIUS server.

The network topology is as follows:

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

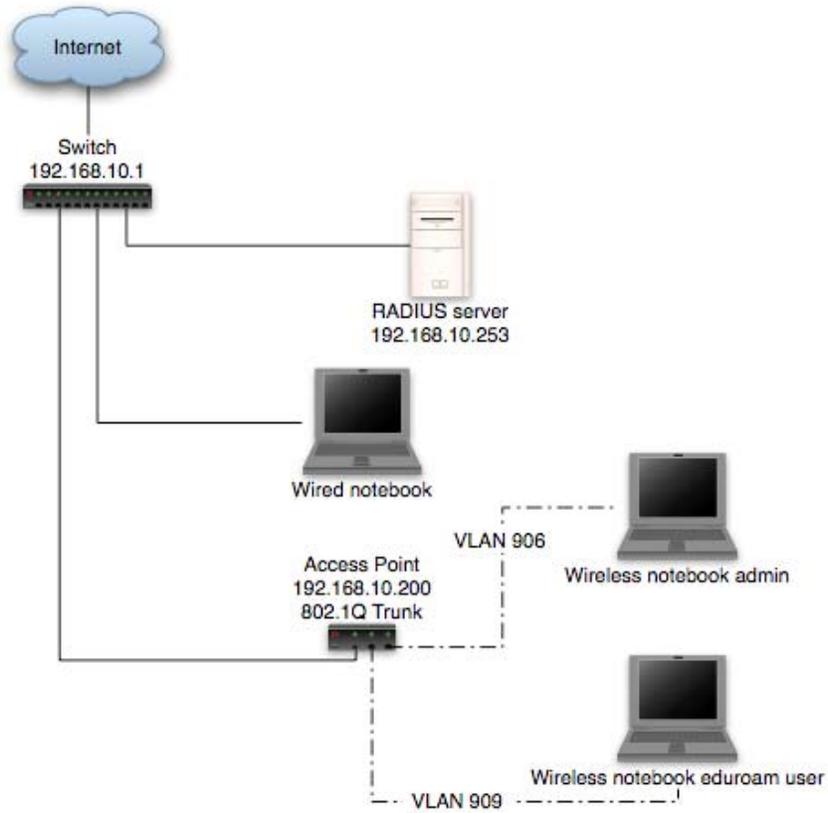


Figure 3.1: Network Topology

In this setup, wireless users are separated in different VLANs: VLAN906 for administrative users and VLAN909 for normal eduroam users. The next table describes each VLAN used in this document:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

VLAN ID	Propose
901	VLAN for internet access – access to core routers
902	The Administrative VLAN of the hotspot (AP's; RADIUS; etc.)
903	VLAN with open SSID for giving information about the institute
906	VLAN reserved for administrative users
909	VLAN reserved for 'normal' eduroam users

Table 3.1: VLAN description

The next table describes the IP configuration for the router sub-interfaces and what networks are configured for each VLAN:

Interface	802.1Q Tag	Interface IP Address	DHCP Pool	What is accessible in this network
FE0.901	901	Some public IP address	N/A	
FE0.902	902	192.168.10.254	N/A	AP's; RADIUS Server
FE0.906	906	10.9.6.254	10.9.6.0/24	administrators
FE0.909	909	10.9.9.254	10.9.9.0/24	eduroam clients

Table 3.2: Router Configuration

3.2.2 Configuring the Ethernet switch for eduroam

In order to gain access to the Internet the configuration of the Ethernet switch needs to be changed. You must create a VLAN in which the Access Points will be placed, and provide it with the correct IP-address and gateway information. This can be done with the commands described below.

The next table describes the VLAN associated with each Port of the switch and what equipment will be connected to that specific port

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

Port	VLAN configuration (T – Tagged; U – Untagged)	What is connected to it
1	U (902)	RADIUS Server
2-47	U (902) T (909)	Access Points
48	U (901) T (902; 909)	Central Ethernet Switch

Table 3.3: Ethernet Switch Configuration

First configure the port where the RADIUS Server will be connected and put it on the Administrative VLAN:

```
switch(config)#interface fastethernet0/1
switch(config-if)#description RADIUS Server
switch(config-if)#switchport mode access
switch(config-if)#switchport access vlan 902
switch(config-if)#spanning-tree portfast
```

Then configure all switch-ports that will connect Access Points for the VLAN's that users and Access Points can have access to (in trunk mode). At a minimum configure the administrative VLAN and the VLAN where authenticated users will be placed:

```
switch(config)#interface range fastethernet0/2 - 47
switch(config-if)#description eduroam Access Points
switch(config-if)#switchport trunk encapsulation dot1q
switch(config-if)#switchport trunk native vlan 902
switch(config-if)#switchport trunk allowed vlan 902, 909
switch(config-if)#switchport mode trunk
```

The uplink can be defined with:

```
switch(config)#interface fastethernet0/48
switch(config-if)#switchport trunk encapsulation dot1q
switch(config-if)#switchport trunk native vlan 901
switch(config-if)#switchport trunk allowed vlan 901, 902, 909
switch(config-if)#switchport mode trunk
```

3.2.3 Setting up the RADIUS server

Now the RADIUS server must be configured.

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

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

are outside the scope of this document).

There is also a bootable CD available based on Knoppix that runs TinyCA, the roCA (read-only CA) that can be found at <http://www.intrusion-lab.net/roca/>.

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.

3.2.3.1 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.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

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 then 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.

3.2.3.2 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.

3.2.3.3 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

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

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.

3.2.3.4 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.)

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
<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=instituion.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

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

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.

3.2.3.5 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, since TTLS is not supported out of the box in the Microsoft Windows environment, therefore third party supplicant needs to be installed by users in case of a TTLS deployment.

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.

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

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

```
<Handler TunnelledByTTLS=1, Realm=tunneled.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>
    # the %D/users file can be empty, it's there for normal PAP
    # authentication. This can however be used for the WEB captive
    # portals.
    Filename %D/users
    EAPType TTLS, PEAP
    EAPTLS_CAFfile %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>
```

3.2.4 RadSec: Using radsecproxy as an add-on to an existing RADIUS server

3.2.4.1 Goals and Prerequisites

This section describes a trimmed-down installation of radsecproxy which enables it to augment any RADIUS server with a RadSec transport. More precisely, it will force a server to:

- Accept requests from a RADIUS server running on the same host.
- Unconditionally forward the requests via RadSec to an upstream server (typically a FLRS or a regional proxy server).
- Accept requests via RadSec from the upstream server.
- Unconditionally forward these requests to a RADIUS server running on the same host.

The prerequisites for this deployment are

- radsecproxy version 1.1 or higher
- The local RADIUS server's DEFAULT realm is configured to forward requests to the radsecproxy port on localhost.
- The local RADIUS server has configured localhost as a client (this is typically the case).
- The deployment requires a server certificate and a private key for that certificate to establish the RadSec connection which designates the server as a service- and identity provider:
(urn:geant:eduroam:component:sp:Europe:<FedName>:<id>
urn:geant:eduroam:component:idp:Europe:<FedName>:<realm>).

3.2.4.2 Sample configuration

Most of the radsecproxy configuration file is static. Therefore, a template configuration file is provided at <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>. A detailed explanation of this configuration file follows, but if you are unwilling to read it all, the comments should make the configuration file almost self-explanatory and you can start and experiment with it right after the initial installation, which is explained below.

3.2.4.3 Installation

1. On UNIX-like systems, the installation is very simple. The code can be downloaded at <http://software.uninett.no/radsecproxy/>. After unpacking it into its base directory, descend into the unpacked directory and type
`make`

The code is ISO C and should compile cleanly. It usually does not require a `./configure`.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

2. After compiling, the executable:radsecproxy is located in the base directory.
Either run this executable here or copy it to a convenient location (e.g. /usr/local/bin) and run it there.
Execution does not require root rights. It expects its configuration file to be in:

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

3. Create the directory /etc/radsecproxy.d/certs/ca/. The template configuration file requires this directory: to contain the accredited CA root certificates and the corresponding Certificate Revocation Lists (CAs) in their OpenSSL hash form.
4. Currently, two CAs are accredited for eduroam. These two certificates and their CRLs need to be copied into this directory. For your convenience, we have prepared a script at <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip> which you can execute in the directory. It will download and hash the appropriate files. In this case you can skip steps 7 and 8.
5. If you want to download the files manually, download them from these URLs:
 - <http://sca.edugain.org/cacert/eduGAINCA.pem>
 - <http://sca.edugain.org/cacert/eduGAINSCA.pem>
 - <http://pki.edugain.org/edugainca/crl/cacrl.pem>
 - <https://sca.edugain.org/crl/cacrl.pem>
6. After downloading, execute the following command in the /etc/radsecproxy.d/certs/ca/ directory:
`c_rehash .`
(including the dot) to generate OpenSSL hashes of the downloaded files.
7. Make sure that the CRL files are regularly re-downloaded. This can for example be done by executing the download script regularly with cron. CRLs have a limited validity time themselves – not regularly refreshing the CRLs will eventually cause certificate validation to fail!
8. Copy the template configuration file into /etc/radsecproxy.conf.
9. Fill the lines marked with __STUFF__.
10. Start radsecproxy and enjoy (for first-time use, starting it with the -f option is recommended, it will start radsecproxy in the foreground and show some verbose startup messages).

3.2.4.4 Detailed explanation of configuration

This section goes through the template radsecproxy.conf line by line and explains the meaning of each stanza.

```
ListenUDP          localhost:11812
ListenTCP          *:2083
```

Since there is a RADIUS server on the same host that occupies UDP/1812, radsecproxy has to listen on a non-standard port. It only needs to listen on the loopback device since it will only communicate with the RADIUS server on the same machine. The choice of 11812 is arbitrary and can be adapted if that port is in use. Since radsecproxy will also accept requests from an upstream RadSec-enabled server, it listens on the default TCP port for RadSec (2083) for requests from outside (the * meaning: all interfaces). If you want radsecproxy to listen only on specific interfaces, enter the interface names here. Beware: in this case you may also have to set the more exotic option SourceTCP (see the man page of radsecproxy for details).

```
LogLevel                3
LogDestination          file:///var/log/radsecproxy.log
```

A logging level of 3 is the default and recommended log level. Radsecproxy will then log successful and failed authentications on one line each. The log destination is a typical POSIX-compliant log location.

```
LoopPrevention          On
```

This enables a semi-automatic prevention of routing loops for RADIUS connections. If you connect a client {...] and a server {...] and give them the same descriptive name, the proxy will prevent proxying.

```
tls default {
    CACertificatePath      /etc/radsecproxy/certs/ca/
    CertificateFile        /etc/radsecproxy/certs/___CERT_PEM___
    CertificateKeyFile     /etc/radsecproxy/certs/___CERT_KEY___
    CertificateKeyPassword ___CERT_PASS___
    CRLCheck              On
}
```

This section defines which TLS certificates should be used by default. This installation of radsecproxy always uses the same certificates, so this is the only TLS section. CACertificatePath contains the eduroam-accredited CA certificates with filenames in the OpenSSL hash form. The parameters below need to be adapted to point to your server certificate in PEM format, the private key for this certificate and the password for this private key if needed, respectively. If no password is needed for the private key, you can comment this line (i.e. precede it with a # sign). The option CRLCheck validates certificates against the Certificate Revocation List (CRL) of the CAs. It requires a valid CRL in place, or else the certificate validation will fail. Therefore, it is important to regularly update the CRLs by re-downloading them as described above.

```
rewrite defaultclient {
    removeAttribute      64
    removeAttribute      65
    removeAttribute      81
}
```

This section deletes attributes from RADIUS requests that convey VLAN assignment information. If VLAN information is sent inadvertently, it can cause a degraded or non-existent service for the end user because they might be put into the wrong VLAN. If you do have a roaming use for cross-institution VLAN assignment, you may want to delete this stanza.

```
Client localhost {
    host 127.0.0.1
        type      udp
        secret    ___LOCAL_SECRET___
}
```

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
client ::1 {  
    type udp  
    secret __LOCAL_SECRET__  
}
```

To accept requests from the RADIUS server on localhost, this server needs to be listed as a client. That server may either use IPv4 or IPv6 for its communication, so both variants are defined. If your system is not IPv6-enabled, simply delete the stanza about **client ::1**. The `__LOCAL_SECRET__` must match the secret which you configured in your RADIUS server for the server catering the DEFAULT realm.

```
client __RADSEC_PEER__ {  
    type TLS  
    secret mysecret  
}
```

To accept requests from the upstream RadSec-enabled server, this other server needs to be listed. Replace `__RADSEC_PEER__` with the hostname of your upstream server. The traditional RADIUS **shared secret** has no meaning in RadSec and can be statically set without security implications. In current eduroam deployments, it is set to **mysecret**. This may change in the future, but will be announced in due time ahead.

```
server localhost {  
    type UDP  
    port 1812  
    secret __LOCAL_SECRET__  
}
```

To deliver requests to the local RADIUS server, this server needs to be configured. See above for the parameter `__LOCAL_SECRET__`.

```
server __RADSEC_PEER__ {  
    type TLS  
    secret mysecret  
    statusserver on  
}
```

This server needs to be configured to deliver requests to the upstream RadSec-enabled server. See above for the configuration item `__RADSEC_PEER__`.

```
realm /myabc\.com$ {  
    replymessage "Misconfigured client: default realm of Intel PRO/Wireless  
supplicant!"  
}  
  
realm /^[^@] {
```

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
        replymessage "Misconfigured client: empty realm!"  
    }  
}
```

There are some known client-side misconfigurations. If they were not already caught by the local RADIUS server, it does not make sense for the proxy to send these requests up to the eduroam infrastructure. These requests are immediately rejected.

Note: if you need to blacklist an existing realm for some reason, you can follow the myabc.com example, copying and replacing it with the realm to be blacklisted.

```
realm /__OWN_REALM__$ {  
    server localhost  
}
```

Requests that are coming in from upstream and are supposed to be handled by the own RADIUS server are listed in `__OWN_REALM__`. Create multiples of these stanzas if your local server serves more than one realm.

Within the RADIUS infrastructure, it is possible to misconfigure a server so that eternal loops are created, which may ultimately lead to a DoS condition. The basic problem is: assume a RADIUS server with its own realm something.org. Further, assume a user who entered john.doe@sub.something.org. A server that only checks if the realm is **equal** to something.org will send this request upstream to the eduroam infrastructure. The infrastructure will in turn detect that the realm ends in something.org and will send the request back to where it came from, which closes the loop.

The configuration stanza above will terminate all requests that end in `__OWN_REALM__` but which were not caught by previous realm definitions, which prevents the loop. This is achieved by having the client name = localhost and the server name = localhost, and LoopPrevention enabled, which checks for this condition.

```
realm * {  
    server __RADSEC_PEER__  
}
```

3.2.5 Obtaining RadSec certificates

IdP and SP certificates are issued by the NRO. Please contact your NRO for more information on how to obtain a certificate.

3.2.6 Configuring the Access Point for eduroam

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

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

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)

3.2.6.1 *The multiple (dynamic) VLAN assignment*

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 feature is often called 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: "encryption vlan vlanID mode ciphers aes-ccm tkip wep128" and "broadcast-key vlan vlanID change 600 membership-termination capability-change", otherwise the access point will not change the user to the received VLAN.

In the example configuration below there will be no dynamic VLAN assignment.

3.2.6.2 *The eduroam SSID – logical setup*

In the following example the eduroam SSID is configured to accept all cipher modes used today on eduroam enabled institutions: 802.1X/WEP; WPA/TKIP; WPA2/AES.

3.2.6.3 *Cisco Aironet 1200 Series basic configuration*

First some basic eduroam configuration parameters for the Cisco AiroNet Series 1200 (similar for other Cisco Access Points Series) are presented.

Setting the Name and IP address

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

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
```

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
```

Configuring the SSID's

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 SSID's can be configured. An open SSID for giving information about the institution and/or how to connect to the eduroam SSID:

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

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
```

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 fastEthernet 0.902
ap1200(config-subif)#encapsulation dot1Q 902 native
ap1200(config-subif)#bridge-group 1

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

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

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

ap1200 (config)#interface dot11Radio 0.906
ap1200 (config-subif)#encapsulation dot1Q 906
ap1200 (config-subif)#bridge-group 6

ap1200 (config)#interface fastEthernet 0.906
ap1200 (config-subif)#encapsulation dot1Q 906
ap1200 (config-subif)#bridge-group 6

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
```

3.2.7 Supplicant

3.2.7.1 SecureW2

EAP-TTLS has been considered the easiest way to implement eduroam in a large (especially student) community. MS Windows has no built-in support for EAP-TTLS, but it can be added by installing SecureW2, a product from Alfa&Ariss Network Security Solution, and thus enable a large user community for EAP-TTLS application. Installing SecureW2 can pose some security issues, which can be addressed by using a preconfigured distribution.

The main security problems are:

- Allowing users to set up new connections. This is one of the advanced options of SecureW2. When disabled it will prevent users from carelessly accepting a rogue RADIUS server, and thus will prevent the users from sending their credentials to a fake server. However running SecureW2 this way requires the necessary certificates for the home RADIUS server to be preinstalled.
- Installation of certificates. SecureW2 requires the certificates necessary to confirm server authenticity to be installed in a specific certificate store; which is different from the default certificate store used by Windows. The result is that even though the user has the main certificate of their Identity Provider installed, SecureW2 will not be able to use it properly to establish the connection.

The proposed procedure is completely secure (under the assumption that users have previously installed their Identity Provider certificate in the standard Windows certificate store). It also vastly simplifies the configuration of SecureW2.

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

Note: Installation instructions for a single SecureW2 client can be found in the appendices.

In order to prepare the preconfigured exe file, the administrator has to take the following steps:

- Prepare the SecureW2.INF file
- Prepare the NSIS configuration file
- Create the exe file with NSIS
- Digitally sign the exe file.

Detailed instructions of these administrative steps, as well as an illustration of the installation process from a user perspective, are provided in the following sections.

3.2.7.2 Preparation of the SecureW2 installer by the administrator

Prerequisites

- 1 Nullsoft Scriptable Install System (NSIS) – Download NSIS from <http://nsis.sourceforge.net/> and install it;
- 2 Code signing
 - a) Download the .NET Framework SDK from Microsoft and install it;
 - b) Prepare a code-signing key using openssl
 - Generate the key
`openssl genrsa -des3 -out test_sign.key 1024`
 - and certification request
`openssl req -new -key test_sign.key -out test_sign.csr`
 - c) The CA needs to certify the key for code signing. Using OpenSSL this is done by
`openssl ca -out test_sign.crt -in test_sign.csr -extensions \ id_kp_codeSigning -extfile codesigning`
 - where the codesigning file contains:
`[id_kp_codeSigning]
extendedKeyUsage = 1.3.6.1.5.5.7.3.3`
 - d) Generate the PFX file
`openssl pkcs12 -export -out test_sign.pfx -in test_sign.crt -inkey \ test_sign.key -clcerts`
 - e) Install the PFX in Windows by double-clicking and accepting the default values except for the key protection where strong protection should be used

Step 1. Prepare the SecureW2.INF file containing:

```
[Version]
Signature = "$Windows NT$"
Provider = "Alfa & Ariss"
Config = 7

[Certificates]
Certificate.1 = your_ca_cert.der

[WZCSVC]
Enable = AUTO

[SSID.1]
```

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
Name = "eduroam"  
Profile = "DEFAULT"  
[Profile.1]  
AuthenticationMethod = PAP  
EAPType = 0  
Name = "DEFAULT"  
Description = "Enter your login credentials:"  
UseAlternateIdentity = FALSE  
VerifyServerCertificate = TRUE  
PromptUserForCredentials = FALSE  
TrustedRootCA.0 = your_ca_cert_fingerprint  
UserName = PROMPTUSER
```

Notes:

- Under the Certificates section you need to provide the entire certification path, starting from the CA directly certifying your RADIUS server and ending with the root CA. Numbering goes from 1 up. As Certificate.0 you may (but do not have to) install the certificate of your RADIUS server.
- As TrustedRootCA.0 you should provide a fingerprint of one of the CAs in the chain (the most natural choice will be the fingerprint of your organisation's CA)
- Section WZCSVC causes the automatic start of Window Zero-Configuration service which is needed for SecureW2
- There are many other options that you may put into the INI file (see SecureW2 administrators guide), but the provided example is sufficient for our configuration. One feature that one might want to add is the use of anonymous outer identity. The configuration above makes the outer identity equal to the real username.

Step 2. Prepare the NSIS configuration directory containing:

1. The configuration file – SecureW2.NSI:

```
;-----  
!define APPLICATION "SecureW2 installer"  
!define VERSION "1.0.0"  
;-----  
!include "MUI.nsh"  
;General  
;Name and file  
Name "${APPLICATION} ${VERSION}"  
OutFile "SecureW2_312_Test.exe"  
;-----  
;Interface Settings  
!define MUI_ICON "your_icon.ico"  
!define MUI_UNICON "your_icon.ico"  
!define MUI_ABORTWARNING  
Section "${APPLICATION}" SecInstall  
SectionIn RO  
; Extract all file to the temp dir  
SetOutPath $TEMPDIR  
; Define all the files required for the installation here:  
File "SecureW2_312.exe"  
File "SecureW2.INF"  
File "your_ca_cert.der"  
ExecWait "SecureW2_312.exe"  
; If an error occurs then goto Error label else goto Continue label  
IfErrors Error  
Goto Continue  
; Error Label, show error box and then quit  
Error:  
MessageBox MB_OK|MB_ICONEXCLAMATION "SecureW2 installation problem, please report to  
..."  
; Continue Label
```

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
Continue:
; Remove temporary files
Delete "$TEMPDIR\SecureW2_312.exe"
Delete "$TEMPDIR\SecureW2.INF"
Delete "$TEMPDIR\your_ca_cert.der"
Quit
SectionEnd
```

Note: you should customise the OutFile name, the text in the MessageBox, the icon filename and the certificate filename(s)

- 2.The original SecureW2 exe file – SecureW2_312.exe
- 3.The SecureW2.INF file
- 4.The icon file (your_icon.ico) – it is advisable to have your own Windows icon to mark your customised installer;
- 5.The certificate file “your_ca_cert.der” (if your certificate chain consists of more files then you need to provide all files listed in the SecureW2.INF file and list them in the SecureW2.NSI file, remember to add them to the Delete section as well);

Step 3. Prepare your customised installer

Right click on SecureW2.NSI and chose “Compile NSI script”. If all went well you should now have your exe file. If there were some errors, NSI will report them.

Step 4. Sign your customised installer

```
C:\Program Files\Microsoft.NET\SDK\v2.0\Bin\signtool sign /a your_installer.exe
```

You can also use the GUI to signtool:

```
C:\Program Files\Microsoft.NET\SDK\v2.0\Bin\signtool signwizard
```

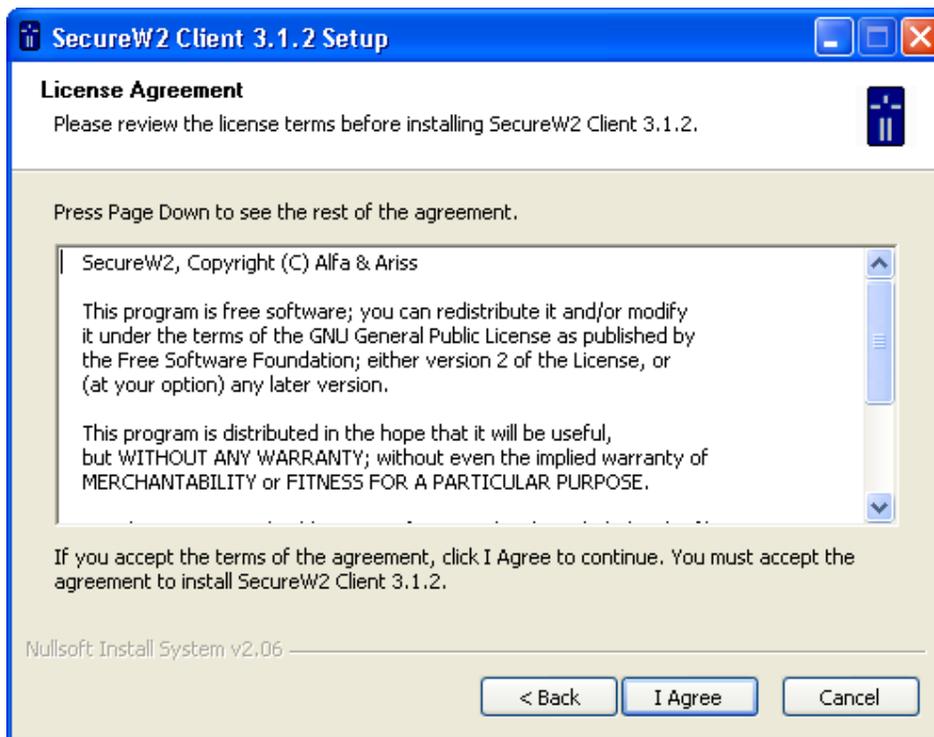
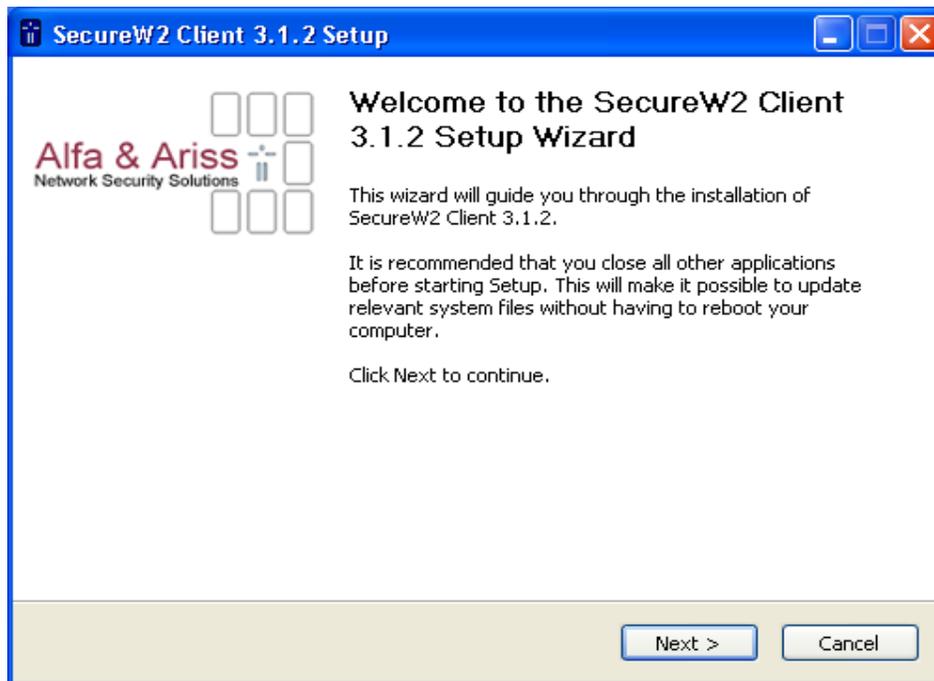
3.2.7.3 User installing SecureW2

The steps for the user are:

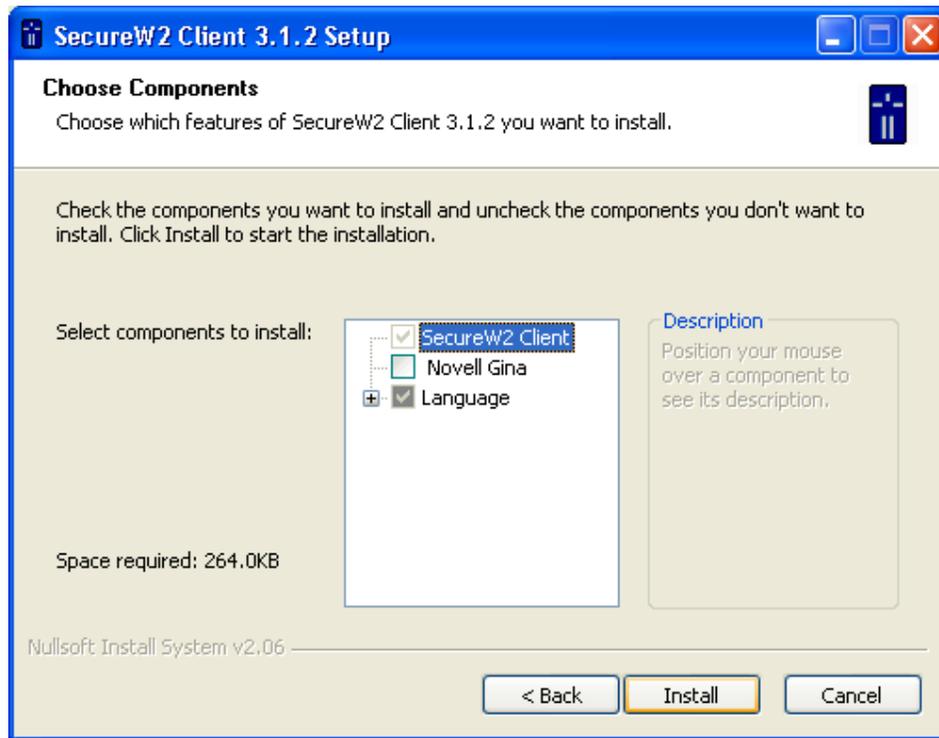
- Download the preconfigured SecureW2 exe file.
- Confirm the signature of the exe file.
- Start the exe file and enter his/her credentials into the program prompt window.
- Reboot the computer.
- Choose SecureW2 as the authentication method for the eduroam network.
- Connect to eduroam.

After downloading the installer, right-click the filename and check the properties for the correct digital signature. After confirming that the file is genuine, start the installer. This immediately starts the SecureW2 installer, which goes through a few self-explanatory steps:

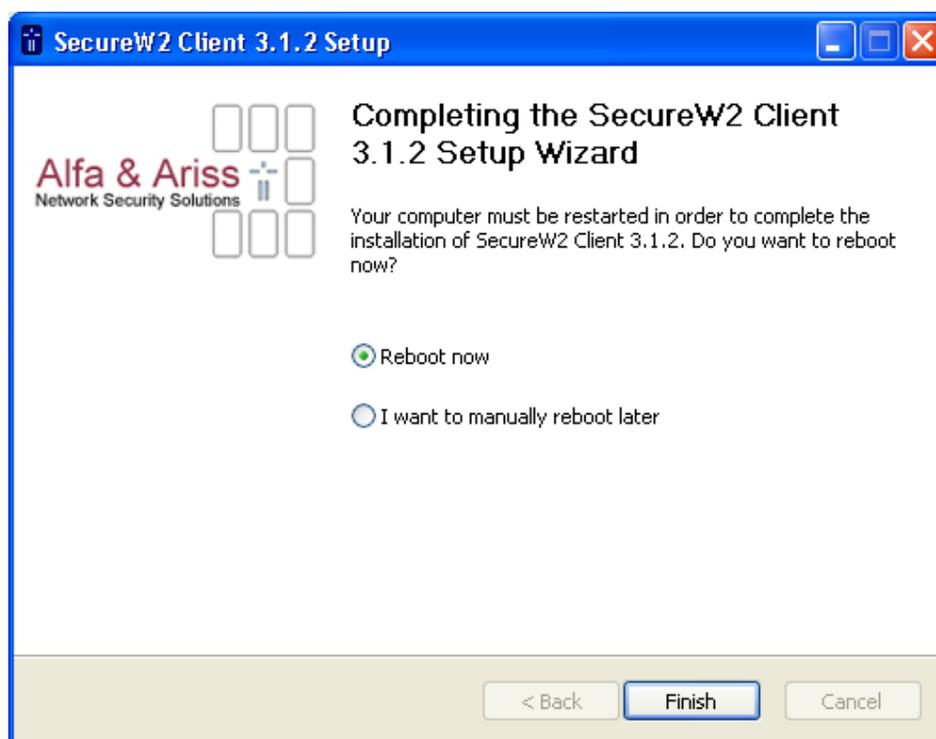
Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



3.3 Operating the Infrastructure

3.3.1 Regular safety precautions

Everyone is responsible for their own network. It is not in the scope for this document to describe regular network or system security measurements. Local “best current practices” for the network and system design should be followed in any case.

3.3.2 Tracking malicious users

If the requirements of the policy are followed, it should be possible to provide sufficient evidence to government agencies to allow them to pinpoint a malicious user, thereby protecting the service provider. The requirements are:

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

- Keeping authentication logs.
- Keeping a DHCP (or even better an ARP) sniffing log.
- Keeping clocks on time with NTP.

The tracing chain is:

1. IP of logged in user that is operating in a malicious manner.
2. IP is linked to MAC address [DHCP or ARP sniffing].
3. Authentication session of this MAC is checked in logs [auth logs].
4. timestamp of authentication and realm is extracted.

From this chain the realm of the offender and the time of login are known. This should be provided to government agencies when required. For example, the information could be: the malicious user is someone from restena.lu who logged in at April 1, 2007, 12:01:45.3221.

Note well: the eduroam service definition document only obliges the Identity Provider and FLR infrastructure to keep logs of all authentications that took place and maintain a synchronised time source. It is in the best interest of a Service Provider to keep sufficient log information themselves though to make sure that the link in steps 3 and 4, above, can be made to trace a user login efficiently. However, even if the SP does not keep the data, it is possible to retrieve the information from FLR logs if need be.

The eduroam operations team can link the realm to a physical point of contact (home federation operator). This federation contact can find the institution, and then in turn the user name of the offender, using the IdP logs with the precise login timestamp that was extracted above.

Note: The User-Name attribute may be obfuscated with an anonymous or even forged outer identity, so it can't be used to reliably identify the individual on the service provider's side. The only reliable User ID is with the IdP.

For the record, the points of the eduroam service definition deliverable DS5.1.1 in question are:

6.2.1: technical contact for federation

6.3.1, Confederation member servers, Bullet 5: logging of authentication attempts

6.3.1, Confederation member servers, Bullet 2: reliable time source

6.3.2, Service Providers, Bullet 2: sufficient layer-2 to layer-3 logging information

6.3.2, Identity Providers, Bullet 4: logging of authentication attempts

Formally, some of these requirements are signed by the federation but affect the institutions, and so should also be re-iterated in the national policies to make them binding for participating institutions. A formal nomination of technical contacts for each institution within a federation is also recommended. For example, Appendix B in DJ5.1.3.2, the national federation policy BCP therefore has sections:

- 4: Logging
- 6.2: Security contact nomination

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

4 Conclusions

This document aims at providing administrators and qualified users with enough technical information to implement eduroam.

However, campus and institution architectures vary widely. Therefore, it is not appropriate to recommend any one specific configuration or vendor. Therefore, this document provides both a basic “recipe” for a typical eduroam setup with equipment that is known to be common at many campuses in the research and education networking community in Europe, as well as giving sufficient information for those using different architectures or vendors to understand the necessary steps to become eduroam-enabled.

To further assist administrators, the appendices that follow contain example configurations for many products as assembled by those in the NREN community in Europe.

It must also be understood that this document will be constantly adapted and augmented as new configurations and implementations become recognised, thereby enabling administrators to keep up-to-date with new trends.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

5 References

- [GN2DJ511]** JRA5 Glossary of terms
<http://intranet.geant2.net/server/show/conMediaFile.6254>
- [GN2DJ512]** Documentation on GÉANT2 Roaming Requirements
<http://www.geant2.net/upload/pdf/GN2-05-71v6.pdf>
- [GN2DJ513,2]** D.Simonsen, J.Rauschenbach, J.Howlett, R.Papez, R.Castro, T.Wiberg, K.Wierenga, S.Winter, et al.
Roaming Policy and Legal Framework Document – Part2. GEANT2 Deliverable DJ5.1.3,2.
November 2006. http://www.geant2.net/upload/pdf/GN2-06-080v5-Deliverable_DJ5-1-3_2_Roaming_Policy_and_Legal_Framework-Part2_20061108094807.pdf
- [GN2DJ514]** K.Wierenga, S.Winter, R.Arends, R.Castro, P.Dekkers, H.Eertink, L.Guido, J.Leira, M.Linden, M.Milinovic, R.Papez, A.Peddemors, R.Poortinga, J.Rauschenbach, D.Simonsen, M.Sova, M.Stanica et al.
Inter-NREN Roaming Architecture: Description and Development Items. GEANT2 Deliverable DJ5.1.4. September 2006. http://www.geant2.net/upload/pdf/GN2-06-137v5-Deliverable_DJ5-1-4_Inter-NREN_Roaming_Technical_Specification_20060908164149.pdf
- [RFC2865]** Remote Authentication Dial In User Service (RADIUS)
<http://www.ietf.org/rfc/rfc2865.txt>

6 Acronyms

Acronyms and terms commonly used in eduroam can be found in the JRA5 Glossary of Terms [GN2DJ511]. Additional terms are listed below.

ARP	[The Address Resolution Protocol (ARP) is the method for finding a host's hardware address when only its network layer address is known. Due to the overwhelming prevalence of IPv4 and Ethernet, ARP is primarily used to translate IP addresses to Ethernet MAC addresses. It is also used for IP over other LAN technologies, such as Token Ring, FDDI, or IEEE 802.11, and for IP over ATM.
BVI	Bridge Virtual Interface is a virtual interface within the campus switch router that acts like a normal <i>routed</i> interface. A BVI does not support bridging, but it actually represents the corresponding bridge group to routed interfaces within the switch router. The interface number is the link between the BVI and the bridge group.
Credentials	A proof of qualification, competence, or clearance that is attached to a person, and often considered an attribute of that person. In this document it refers to username/password pairs.
FQDN	Fully Qualified Domain name is an unambiguous domain name that specifies the node's position in the DNS tree hierarchy absolutely. To distinguish an FQDN from a regular domain name, a trailing period is added. Ex: somehost.example.com. An FQDN differs from a regular domain name by its absoluteness; a suffix will not be added]
GUI	Graphical User Interface is a particular case of user interface for interacting with a computer which employs graphical images and widgets in addition to text to represent the information and actions to the user. Usually the actions are performed through direct manipulation of the graphical elements.
IANA	Internet Assigned Numbers Authority is broadly responsible for the allocation of globally-unique names and numbers that are used in Internet protocols that are published as RFC documents. It maintains a close liaison with the IETF and RFC Editor in fulfilling this function.
MPPE	Microsoft Point-to-Point Encryption is a protocol for encrypting data across Point-to-Point Protocol and Virtual Private Network links. It uses the RSA RC4 encryption algorithm. MPPE supports 40-bit, 56-bit and 128-bit session keys, which are changed frequently to improve security. The exact frequency at that the keys are changed is negotiated, but may as frequent as every packet. MPPE alone does not compress or expand data, but the protocol is often used in conjunction with Microsoft Point-to-Point Compression, which compresses data across PPP or VPN links.
MS-IAS	Internet Authentication Service is the Microsoft Implementation of a Remote Authentication Dial-in User Service (RADIUS) server and proxy. As a RADIUS server, IAS performs centralized connection authentication, authorization, and accounting for many types of network access, including wireless and

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

	virtual private network (VPN) connections. As a RADIUS proxy, IAS forwards authentication and accounting messages to other RADIUS servers.
.NET	Microsoft .NET is an umbrella term that applies to a collection of products and technologies from Microsoft. Most have in common a dependence on the Microsoft .NET Framework, a component of the Windows operating system.
PFX	Personal inFormation eXchange is a common file extension for X.509 certificates. A .pfx File may contain certificate(s) (public) and private keys (password protected).
NRO	National Roaming Operator is an organisation responsible for coordinating roaming in their designated area.
PID	The process identifier (normally referred to as the process ID or just PID) is a number used by some operating system kernels (such as that of UNIX or Windows NT) to uniquely identify a process. Under Unix, the PID of a newly created child process is returned by the fork() system call to the parent. The PID can be passed to process control functions like waitpid() or kill() to perform actions on the given process, and if the operating system as procs support the files in /proc/pid/ can be queried for information about the process.]
SDK	A Software Development Kit (or “devkit”) is typically a set of development tools that allows a software engineer to create applications for a certain software package, software framework, hardware platform, operating system or similar. It may be something as simple as an “application programming interface” in the form of some files to interface to a particular programming language, or include sophisticated hardware to communicate with a certain embedded system. Common tools include debugging aids such as an IDE and other utilities. SDKs also frequently include sample code and supporting technical notes or other supporting documentation to help clarify points from the primary reference material.
SA5	Service Activity 5 is the GÉANT2 Service Activity dedicated to developing eduroam as a service. Established in September 2007, official members of the eduroam SA are the representatives of those roaming federations that are GN2 project partners and have signed the eduroam policy.
SNMP	Simple Network Management Protocol forms part of the Internet protocol suite as defined by the Internet Engineering Task Force (IETF). More specifically, it is a Layer 7 or Application Layer protocol that is used by network management systems for monitoring network-attached devices for conditions that warrant administrative attention.

Appendix A RADIUS servers

The main part of the document covered the configuration of the Radiator RADIUS implementation for the various servers in the eduroam hierarchy. As this cannot be understood as a recommendation to use this specific implementation, the following parts of Appendix A present the configuration of alternative RADIUS implementations. Starting with Radiator as institutional server only to complement the main part, later on the respective configurations for Freeradius, Navis, Vital AA and MS-IAS are presented.

As the eduroam infrastructure is build upon RADIUS servers, the following picture depicting the message flow inside an institutional RADIUS server might be helpful.

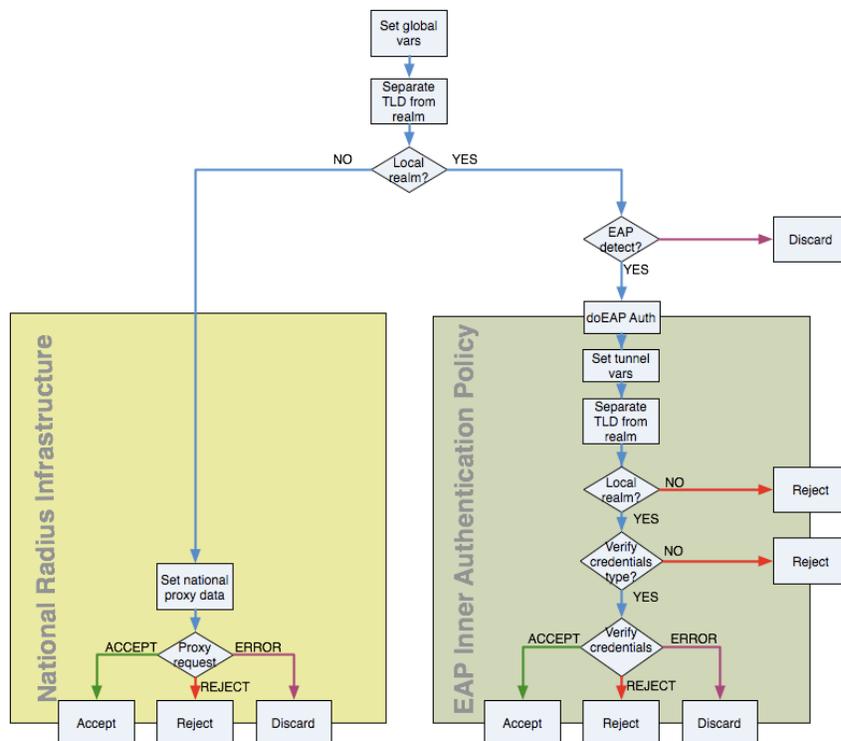


Figure A 1: Message flow in RADIUS server Identity Management System

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

A.1 Radiator institutional server

Further explanation for configuring the Radiator institutional server is given in sections 3.1 “Common FLRS proxy setups” and 3.2.3 “Setting up the RADIUS server”.

An example configuration script can be downloaded from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>.

A.2 FreeRADIUS institutional server

A.2.1 Setting up FreeRADIUS

FreeRADIUS (<http://www.freeradius.org>) is a very powerful, configurable, fast and freely available open-source RADIUS server. The sample configurations are based on FreeRADIUS 2.0.3. RPM installation packages exist for numerous distributions under the name of either *freeradius* or *freeradius-server*. The use of version 1.x is discouraged. If the personal skills allow, it is suggested to download the most recent version from the FreeRADIUS web site [link: www.freeradius.org] and compile it fresh from these sources.

All the configuration files for FreeRADIUS are stored in a */etc/raddb* directory. Configuration is broken down into several files by default.

The most important ones for an eduroam Service Provider are:

- *clients.conf* – Definition of the client devices which connect to the server (see A.2.2)
- *proxy.conf* – Definition of other servers and realms for proxying (see A.2.3)
- *sites-enabled/eduroam* - virtual server definition (see A.2.4)
- *radiusd.conf* – main configuration file (see A.2.5)

For eduroam Identity Providers, additionally the following file is important:

- *eap.conf* – define EAP authentication (see A.2.6)

In this section, an example server is configured as follows:

- Acting as a Service Provider:
Accept requests from Access Points.
Proxy requests to IdPs in the eduroam infrastructure.
- Acting as a Identity Provider:
Usage of both EAP-TTLS/PAP and PEAP/MSCHAPv2 simultaneously for authenticating the clients.

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

- Optionally: forcing outer identity to anonymous@domain.tld.
- A few test user accounts are statically entered in a file.
 - Regular users are authenticated from an LDAP directory, using the eduPerson schema. Usernames are stored in eduPersonPrincipalName attribute and the plaintext password in userPassword attribute.
 - Accounting is stored in a MySQL database.
 - Optionally a script can be set-up to update accounting SQL table with information of assigned IP addresses.

A.2.2 Defining clients - Access Points and RADIUS servers in *clients.conf*

Access points, RADIUS servers and other RADIUS clients (NAS devices, RADIUS test scripts, ...) are defined in the file `/etc/raddb/clients.conf`. This file lists devices that may send requests to the server. An example is given below, and can be downloaded from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>.

This example defines a group of access points with a common shared secret: 158.64.254.0 – 158.64.254.7 (due to netmask = 29) are defined as clients and are assigned the shared secret *verysharedsecret*. These clients will show up in log files with their shortname *eduroam-ap-v4*. All incoming requests by these clients will be handled by the virtual server *eduroam*.

```
client descriptivename {
    ipaddr                = 158.64.254.0
    netmask                = 29
    secret                = verysharedsecret
    require_message_authenticator = no
    shortname              = eduroam-ap-v4
    nastype                = other
    virtual_server         = eduroam
}
```

To edit the file for your own configuration, comment out or delete the existing entries and add the access points:

An arbitrary number of client stanzas can be added to the file. Individual IP addresses with their own shared secret can be defined by assigning *netmask = 32*.

For Cisco APs the *nastype* is set to *cisco*, for other NAS devices the value should be set differently. If the AP has no corresponding *nastype*, the *nastype* needs to be set to „other“. It is recommended to use a different shared secret for every access point. The secret has to be entered in the Access Point as well. You can use

`mkpasswd` to randomly generate shared secrets. Shared secrets in RADIUS are rather weak; for cryptographic reasons, the recommend length for shared secrets is 16 bytes.

Since a linked RADIUS server is viewed as a RADIUS client device, they also have to be added here. Hence it is necessary to determine all servers DNS names or their IP addresses. For an institutional server, at least its uplink RADIUS servers from the federation have to be added (in most cases, these are the Federation-Level RADIUS servers (FLRS)). client stanzas can also define IPv6 clients. The example files found in <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip> show two FLRS uplinks (Luxembourg FLRS servers), one for IPv6 and one for IPv4 clients.

A loopback client is useful for running testing scripts and even mandatory for tunnelled authentication methods like TTLS and PEAP, so we make sure it is set correctly. The localhost's secret does not need to be shared with anyone, it is just there proforma and can even be left at the default „testing123“ An example can be downloaded from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>.

A.2.3 Configure realm handling and proxying: proxy.conf

RADIUS requests from local users must be handled locally, while requests from roaming users must be proxied to the national FLRS RADIUS server. If an organisation has a domain *domain.tld* all requests for **.domain.tld* are forwarded from the national RADIUS server to their organisational RADIUS server and it is their responsibility to filter out invalid domains via the blackhole rule to prevent looping of invalid authentication requests. Proxying is configured in the file `/etc/raddb/proxy.conf`, the first rule with a match is used. The configuration file contains lots of sample definitions. These are not necessary and can be deleted.

An example file can be downloaded as an example and for edit (<http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>). Points of importance are:

- Replace `domain.tld` for your own assigned realm as an IdP. For SPs, only the DEFAULT realm is important.
- `proxy.conf` contains a small configuration stanza; proxy server for system-wide options:

```
proxy server {  
    default_fallback = yes  
}
```

This setting means that if an authentication request contains a realm that is not explicitly listed in later sections of this file, it is proxied to the DEFAULT realm.

- The configuration file also needs to have defined `home_server` entries. They are the counterpart to the client stanzas from above: other RADIUS servers with their IP addresses and shared secrets. Two such example stanzas (for FLRS proxying of unknown realms) are included in the example file.
- The parameters `response_window`, `zombie_period` and `revive_interval` are tuning parameters and

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

should initially remain untouched. Using `status_check = status-server` is recommended, but only if the server on the other end is configured to handle such Status-Server requests. This behaviour is configured in `radiusd.conf`.

- Home servers are aggregated in pools (i.e.: all servers that have the same function, like: all FLRS servers, are grouped together). This happens very straightforward in a `home_server_pool` definition:
- All realms known to the server are listed in `realm` sections.
- The `nostrip` command specifies that no domain stripping is performed (usernames always remain Network Access Identifier in the form [short_username@domain.tld](#), for both local and roaming users).

A.2.4 Virtual server definition for eduroam: sites-enabled/eduroam

For a Service Provider, the definition of the actual server (the part that processes incoming packets from the defined clients and sends them off to the defined proxy servers) is very simple, since the server doesn't have to perform any authentication by itself.

Configuration of a server section happens in its own file within the subdirectory `sites-available`. The file can be downloaded from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>. A symlink in the directory `sites-enabled` points to the definition in `sites-available`.

The file contains a single server section with several subsections. Authentication requests are processed through the following sections, in order:

- `authorize`: Collect information about the user from the RADIUS client.
- `pre-proxy`: Prepare proxy to other server.
- `post-proxy`: Evaluate response from other server.
- `post-auth`: Final packed inspection before sending the answer to the RADIUS client.

Accounting requests are processed as follows:

- `preacct`: Collect information about the session which is being accounted.
- `pre-proxy`: Prepare proxy to other server.
- `post-proxy`: Evaluate response from other server.

These sections contain the names of modules that are configured in `radiusd.conf` and are explained in a later section. The complete server configuration for an SP is below (note the empty `authenticate` section since the server is not actually performing any authentication itself as an SP):

As the script is processed, the following actions are performed:

- `authorize/auth_log`: The request packet is dumped into a log file.
- `authorize/suffix`: The username is examined to determine the realm.
- `authorize/if`: Loop prevention of bogus subdomain routing.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

- pre-proxy/attr_filter.pre-proxy: Filter out attributes which are not supposed to be sent out (VLAN etc.).
- pre-proxy/pre_proxy_log: Log the request to disk before it is proxied.
- post-proxy/post_proxy_log: Log the request to disk after receiving the answer from the proxy.
- post-proxy/attr_filter.post-proxy: Filter out unsolicited attributes from the proxy (VLAN etc.).
- post-auth/reply_log: The final reply to the client is logged to file.

For accounting requests, the following steps are performed:

- preacct/detail: The request packet is dumped into a log file.
- preacct/suffix: The username is examined to determine the realm.
- preacct/if: Loop prevention of bogus subdomain routing.
- pre-proxy/attr_filter.pre-proxy: Filter out attributes which are not supposed to be sent out (VLAN etc.).
- pre-proxy/pre_proxy_log: Log the request to disk before it is proxied.
- post-proxy/post_proxy_log: Log the request to disk after receiving the answer from the proxy.
- post-proxy/attr_filter.post-proxy: Filter out unsolicited attributes from the proxy (VLAN etc.).

A.2.5 Main configuration file: radiusd.conf

This file contains many generic RADIUS configuration items. However, the defaults to these items are usually appropriate for eduroam, and therefore will not be discussed here. Only configuration items that need alteration are discussed.

- The following lines are commented-out by default:

```
user = radiusd  
group = radiusd
```

If the lines remain commented-out, FreeRADIUS will run as root. This is generally considered inadvisable. Therefore, it is strongly recommended that you create a radiusd user and a radiusd group, and then run the server as this radiusd user. However, if you do this you must ensure that that all files the server accesses during operation can be written to and read by this user and group.

- As of version 2, FreeRADIUS is capable of both IPv4 and IPv6. The following four sections enable both authentication and accounting processing with IPv4 and IPv6:

```
listen {  
    type = auth  
    ipaddr = *  
    port = 1812  
}
```

```
listen {
    type = auth
    ipv6addr = ::
    port = 1812
}

listen {
    type = acct
    ipaddr = *
    port = 1813
}

listen {
    type = acct
    ipv6addr = ::
    port = 1813
}
```

- The following lines are important for eduroam operation: the aforementioned possibility to use Status-Server requests is enabled in the *security* section, and all the defined client definitions, proxy server definitions and the virtual servers are read in. The small subset of modules used in the virtual server eduroam are also defined here:

```
security {
    max_attributes = 200
    reject_delay = 0
    status_server = yes
}

proxy_requests = yes
$INCLUDE proxy.conf
$INCLUDE clients.conf
$INCLUDE sites-enabled/
```

- detail modules take a filename as an option to determine where to log packet content. The choice of file name is arbitrary. The %Y%m%d name above gets expanded to a date, and thus creates a new

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

subdirectory every day. This makes debugging and deleting old records very easy. This module shows up in the sample server configuration in several variations (different filenames): `auth_detail`, `detail`, `pre_proxy_detail`, `post_proxy_detail`, `reply_detail`.

```
detail auth_log {
    detailfile = ${radacctdir}/%Y%m%d/eduroam/auth-detail
    detailperm = 0600
}
```

- The following module determines which part of the incoming User-Name is the name, and which one is the realm. Using `@` as delimiter and treating the part behind the `@` as realm is the eduroam definition of user names (and is the suggested naming scheme in RFC4282).

```
realm suffix {
    format = suffix
    delimiter = "@"
}
```

- The following module gives the potential for filtering out unwanted attributes. One example for this is VLAN assignments. These assignments are of local significance only, and therefore it does not make sense for a remote server to dictate VLAN membership. In fact, this may even be harmful, and so, VLAN attributes should be not be sent out by IdPs and also be filtered by SPs. The module refers to files in the main configuration directory. Please refer to these files themselves for additional documentation on how to specify which attributes are allowed and which not. As of FreeRADIUS 2.0.4, the server will ship with standard defaults that are compatible with eduroam operation without the need for modifications.

```
attr_filter attr_filter.pre-proxy {
    attrsfile = ${confdir}/attrs.pre-proxy
}
```

- The remaining parts in the virtual server, *if (...) update { ... }* are not separate modules but a configuration language. Details about usage of this configuration language are available on its man page, see *man unlang*.

A.2.6 User authentication: configuring an eduroam IdP

Configuring an IdP involves:

- Configuring the server as EAP endpoint.
- Configuring the local database backend.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

EAP configuration happens in the file `eap.conf` (downloadable from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>). This example, `eap.conf` file is configured to accept both TTLS and PEAP requests and process them with the same backend. For a discussion of when to use TTLS and when PEAP, please refer to section 3.2.3.5 of this document. The following notes refer to different modules within the file:

- In the `tls` section, the server certificate that is presented to the users while they authenticate is defined. `certdir`, `private_key_...` have to point to the appropriate files on the server. It appears confusing for most new administrators that `tls` needs to be defined even though EAP-TLS is not a desired authentication method. This is because the `ttls` and `peap` sections share the certificate parameters and rely on the existence of the `tls` stanza. Unwanted EAP-TLS authentication can be prevented by pointing `cadir` to a dummy Certification Authority (like the one that is created on first server startup) which never issues proper client certificates.
- The `ttls` and `peap` sections denominate a dedicated virtual server to handle the inner tunnel request (which happens after the TLS handshake with the supplicant has completed). Note that there needs to be an empty `mschapv2` stanza in `eap.conf` to make PEAP work – this is due to internal server workings only.
- To include EAP into the virtual server `eduroam` (see section A.2.4 “Virtual server definition for eduroam: sites-enabled/eduroam”), only two lines need to be added; the new virtual server “`eduroam-inner-tunnel`” (which we defined in `eap.conf` before) needs to be set up to do the actual user authentication, and the module `eap` must be listed at the end of the `authorize` and `authenticate` sections.:

```
server eduroam-inner-tunnel {
    authorize {
        auth_log
        files
        ldap
        mschap
        pap
    }
    authenticate {
        Auth-Type LDAP{
            ldap
        }
        Auth-Type PAP{
            pap
        }
        Auth-Type MS-CHAP{
            mschap
        }
    }
}
```

```
    }
    preacct {
    }
    accounting {
        detail
        sql
    }
    session {
    }
    post-auth {
        reply_log
        Post-Auth-Type REJECT {
            reply_log
        }
    }
    pre-proxy {
    }
    post-proxy {
    }
}
```

The most notable difference between the inner server and the outer is that the inner tunnel does not do EAP. Instead, it uses the *ldap* module in the *authorize* and *authenticate* sections to authenticate users against the ldap backend. Note that the ldap module is configured in *radiusd.conf*. You have to enter the appropriate connection details to your LDAP server there and open firewall ports from the RADIUS server to the LDAP server if need be.

Also note that this virtual server does not call the *suffix* module. This means the inner request does not get proxied elsewhere. This is a standard security measure in eduroam.

The virtual server definition also includes the *files* module. When doing LDAP authentication, this is optional. In the example file, we use this module to have test users in the flat file “users” in order to have a possibility of debugging authentication. Once the authentication against the file works and the LDAP connection is set up properly, the files module is only needed for advanced topics like VLAN assignment.

- For an initial setup of the *users* file, administrators should delete all existing entries and add only the following line in the file:
`test@domain.tld Cleartext-Password := "<test password>"`

This creates a test user with the name “[test@domain.tld](#)” and a password. It can be used to test authentication.

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

- In case a VLAN assignment should be done, the users file needs to contain the corresponding RADIUS attributes, like in the following example for VLAN 313:

```
DEFAULT <assignment conditions here> Tunnel-Type =VLAN,  
Tunnel-Medium-Type =IEEE-802,  
Tunnel-Private-Group-Id =313
```

- The configuration of the ldap module itself happens in radiusd.conf. This config stanza is extensively documented in the file, so below just a simple example of a sample configuration (based on an eduPerson LDAP schema).

```
modules {  
  
    ldap {  
        server = "localhost"  
        identity = "cn=RADIUS,dc=domain,dc=tld"  
        password = "<secret for identity dn>"  
        basedn = "ou=student,dc=domain,dc=tld"  
        filter = "(eduPersonPrincipalName=%{User-Name})"  
        start_tls = no  
    }  
  
    # Mind the Linewrap!
```

- If logging of accounting requests to a database is desired (see below), it is also wise to extend the accounting key in the acct_unique module to make accounting more robust.

```
acct_unique {  
    key = "User-Name, Acct-Session-Id, Calling-Station-Id,\  
        Called-Station-Id, NAS-IP-Address, NAS-Port"  
}
```

A.2.7 Setting up accounting in the SQL database

All accounting information is logged into the MySQL database. The configuration is in the file /etc/raddb/sql.conf. The content of this file should be replaced with that found in <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>.

A.2.8 Logging the client IP address as Service Provider (Optional)

The client IP address is logged in the DHCP log file. However this information can also be stored in the ACCOUNTING table with other accounting data and thus provide easy access to that data. A Sysadmin needs to set-up a script to update the SQL database information with the assigned IP address. The client IP address is determined by tailing the DHCP log file and monitoring all IP assignments. The script also monitors active connections and cleans up accounting (closes accounting) for stale connections. SNMP access to Access Points is required.

The script is available from here:

http://www.eduroam.si/uploads/CN/eC/CNeCC3Uc7XI9Tw_dLtwYZg/eduroam_monitor-20060809.tar.bz2

The access points need to be registered with the script. This is done by entering the needed access point data into the database:

```
USE radius;
create table access_points (
  `IP address` varchar(100) PRIMARY KEY NOT NULL,
  `snmp secret` varchar(100) NOT NULL default '',
  `radius secret` varchar(100) NOT NULL default '',
  `root username` varchar(100) NOT NULL default '',
  `root password` varchar(100) NOT NULL default ''
) TYPE=MyISAM;
```

A.2.9 More information

The original Slovenian Eduroam technical specifications and sample configuration site:

<http://www.eduroam.si>

ARNES AAI technical support e-mail address: aaa-podpora@arnes.si

FreeRADIUS files:

<http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>

Eduroam-in-a-box web configuration tool:

<http://eduroam.sourceforge.net>

A.3 VitalAAA Institutional and national server

A.3.1 Institutional VitalAAA Server (ver. 5.2.0+)

To set-up a local RADIUS server for Institutional VitalAAA, you must change the following configuration files:

- server_properties
- method_dispatch
- clients

Also, you must download the following files from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>:

- error.pf
- proxy.pf
- acct.pf
- prepare.pf
- local.pf

server_properties file

Make the following changes in the `server_properties` file:

```
Radius-Acct-Address = "*:1813"  
Radius-Auth-Address = "*:1812"  
Database-Address = "0"  
Radius-CharSet = UTF8  
Delimiter-Precedence = "@"  
Suffix-Delimiters = "@"
```

method_dispatch file:

Make the following changes in the `method_dispatch` file:

```
radius      Auth 1      prepare      setWorkingVars
```

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
radius          acct 4      acct          dumpAcct
```

clients file

Add the following lines with the national proxy server information in the `clients` file:

```
<192.168.1.10    national_server_secret>
<192.168.1.20    national_server_secret>
...
```

A.3.2 National VitalAAA Server (ver. 5.2.0+)

To set-up of a (national) RADIUS proxy server for VitalAAA you must change the following configuration files:

- `server_properties`
- `method_dispatch`
- `clients`

You must also download the following files from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>:

- `aaa.pf`
- `error.pf`
- `proxy.pf`
- `prepare.pf`

server_properties file:

```
Radius-Acct-Address = "*:1813"
Radius-Auth-Address = "*:1812"
Database-Address = "0"
Radius-CharSet = UTF8
Delimiter-Precedence = "@"
Suffix-Delimiters = "@"
```

method_dispatch file:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

```
radius      Auth  1      prepare      setWorkingVars
radius      acct  4      aaa      dropRadiusAcct
```

clients file

Add the lines with the eduroam proxy server and the local RADIUS servers to the `clients` file:

```
192.87.106.34      <eduroam_secret>
130.225.242.109   <eduroam_secret>
<192.168.1.10>    <local_server_secret>
<192.168.1.20>    <local_server_secret>
...
```

A.4 Microsoft Internet Authentication Service server as institutional server

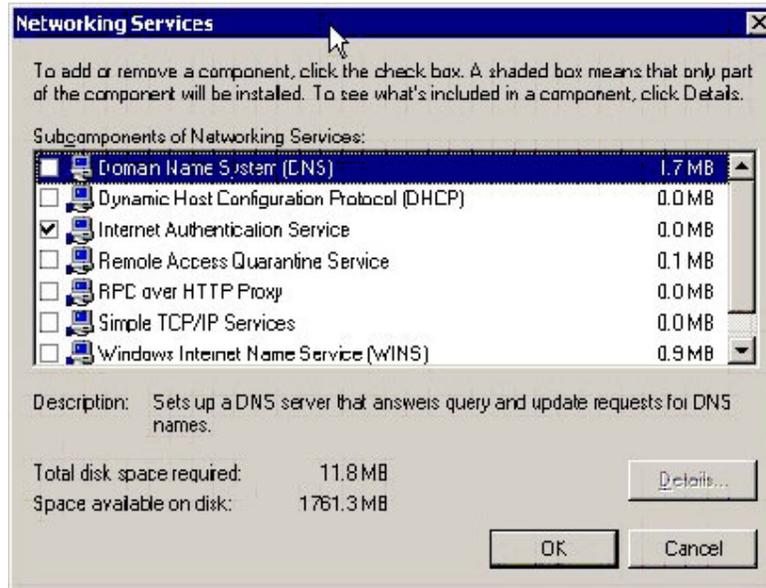
Internet Authentication Service (IAS) in Microsoft Windows Server 2003, Standard Edition; Windows Server 2003, Enterprise Edition; and Windows Server 2003, Datacenter Edition is the Microsoft implementation of a Remote Authentication Dial-in User Service (RADIUS) server and proxy. You can configure IAS in Windows Server 2003, Standard Edition, with a maximum of 50 RADIUS clients and a maximum of 2 remote RADIUS server groups. You can define a RADIUS client using a fully qualified domain name or an IP address, but you cannot define groups of RADIUS clients by specifying an IP address range. In the Enterprise and Datacenter Edition of Windows Server 2003 these limitations do not exist.

A.4.1 Installing IAS

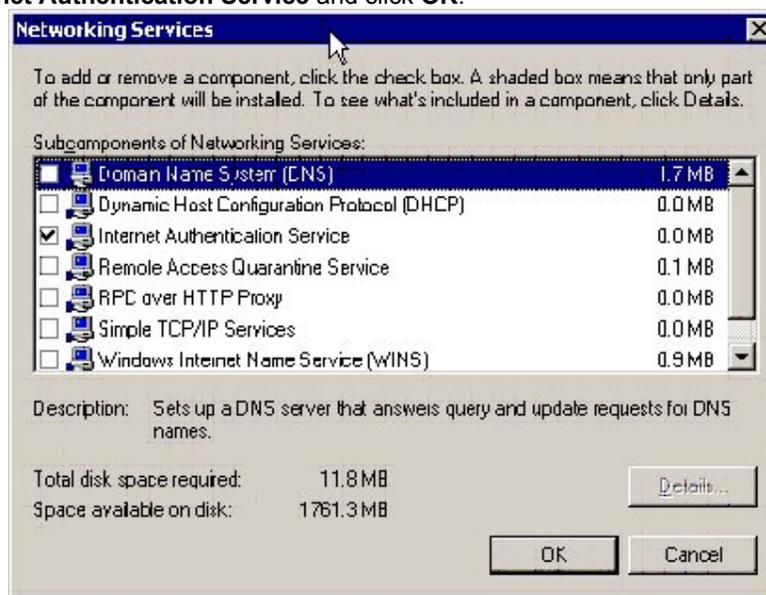
Windows Server 2003 does not install IAS in the default installation. You must install the IAS separately, as follows:

5. Select **Control Panel>Add or Remove Programs>Add/Remove Windows Components>Networking Services:**

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



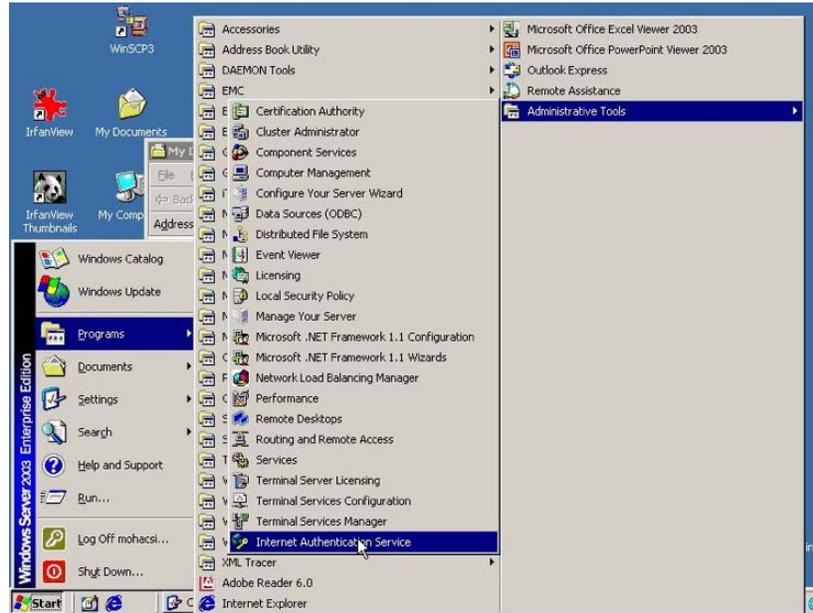
6. Select **Internet Authentication Service** and click **OK**:



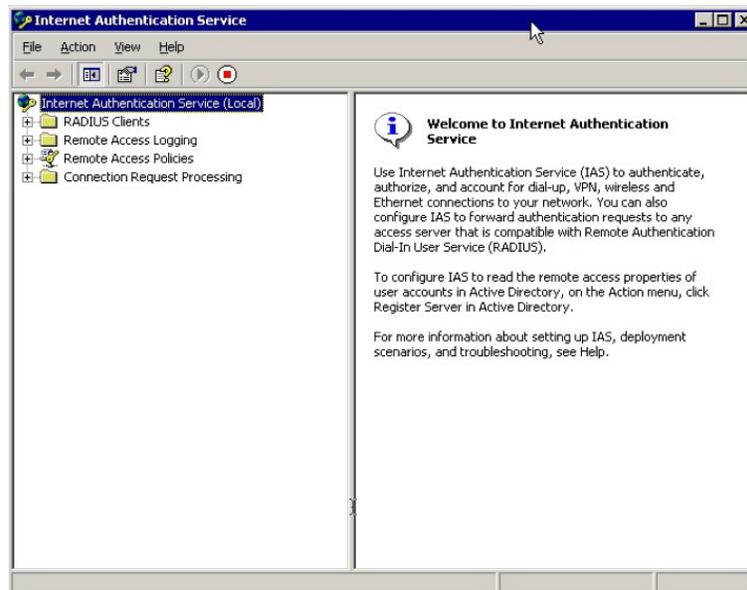
After the installation has finished, you must open the IAS administrative console:

7. Select **Administrative Tools**:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



8. Click **Internet Authentication Service** in the **Start** menu to start the IAS console:

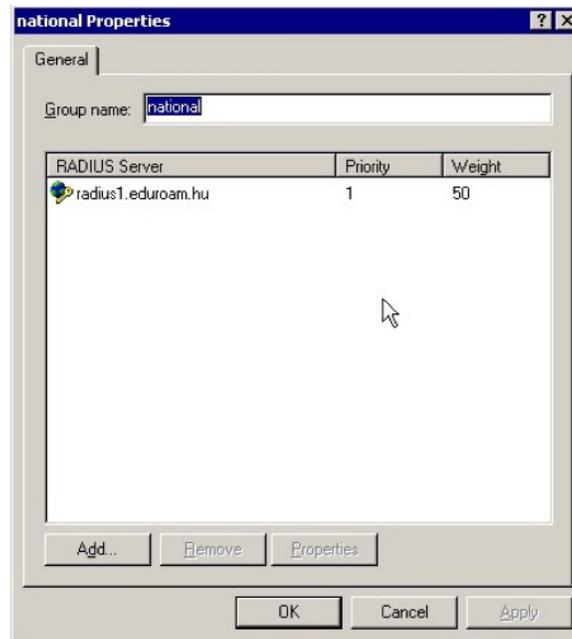


You must then configure IAS as described in the sections that follow.

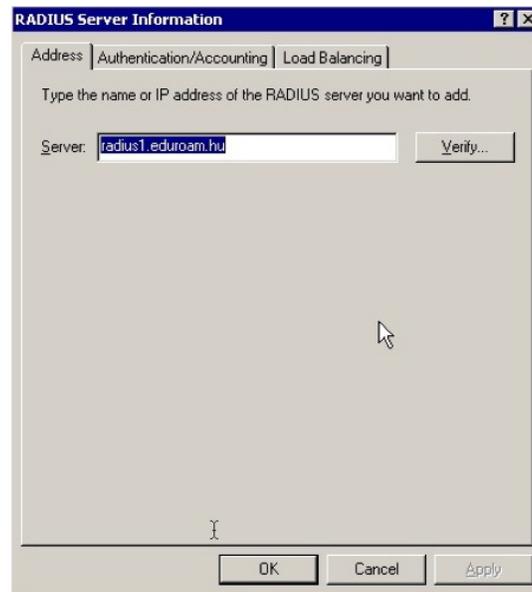
A.4.2 Configuring remote RADIUS servers

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

The national RADIUS proxy server must be added to the remote RADIUS server:

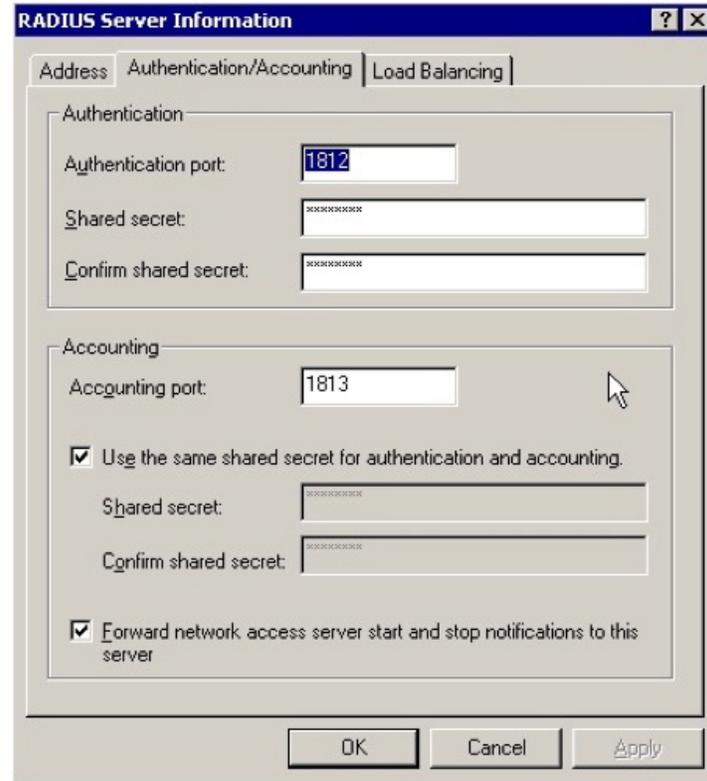


The remote RADIUS server address must be specified:



You have to enter the RADIUS server authentication port (usually 1812) and the shared secret of the remote RADIUS proxy server as well as the remote RADIUS server accounting port. You can specify different shared secrets for accounting if you wish:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



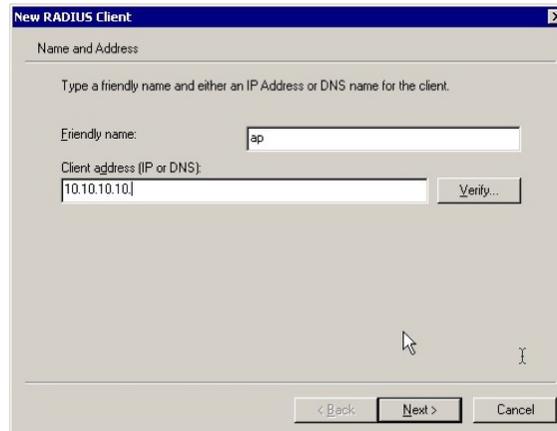
A.4.3 Configuring IAS to act as a university RADIUS server in the eduroam hierarchy

Configuring IAS for access points and upstream proxies

For each access point and upstream proxy (i.e. national eduroam RADIUS server) the parameters of the RADIUS Clients must be configured:

When you add a new access point, a wizard will appear requesting the name and the IP address of the RADIUS client (i.e. Access Point, switch, or upstream RADIUS proxy):

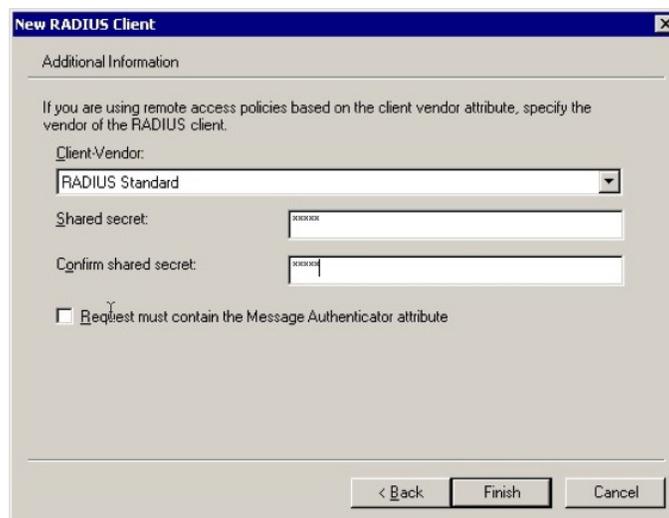
Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



The dialog box is titled "New RADIUS Client" and has a tab labeled "Name and Address". Below the tab is a text area with the instruction: "Type a friendly name and either an IP Address or DNS name for the client." There are two input fields: "Friendly name:" with the value "ap" and "Client address (IP or DNS):" with the value "10.10.10.10". A "Verify..." button is located to the right of the second field. At the bottom of the dialog are three buttons: "< Back", "Next >", and "Cancel".

You must then:

- Specify the shared secret between the RADIUS client and your RADIUS server (IAS):



The dialog box is titled "New RADIUS Client" and has a tab labeled "Additional Information". Below the tab is a text area with the instruction: "If you are using remote access policies based on the client vendor attribute, specify the vendor of the RADIUS client." There is a "Client-Vendor:" dropdown menu with "RADIUS Standard" selected. Below it are two text fields: "Shared secret:" and "Confirm shared secret:", both containing masked characters. A checkbox labeled "Request must contain the Message Authenticator attribute" is unchecked. At the bottom of the dialog are three buttons: "< Back", "Finish", and "Cancel".

You can select various vendors of RADIUS clients, but in most of the cases you should use **RADIUS Standard**.

Configuring Connection Request Processing Policy

The realm processing should be configured to match eduroam hierarchy, that is:

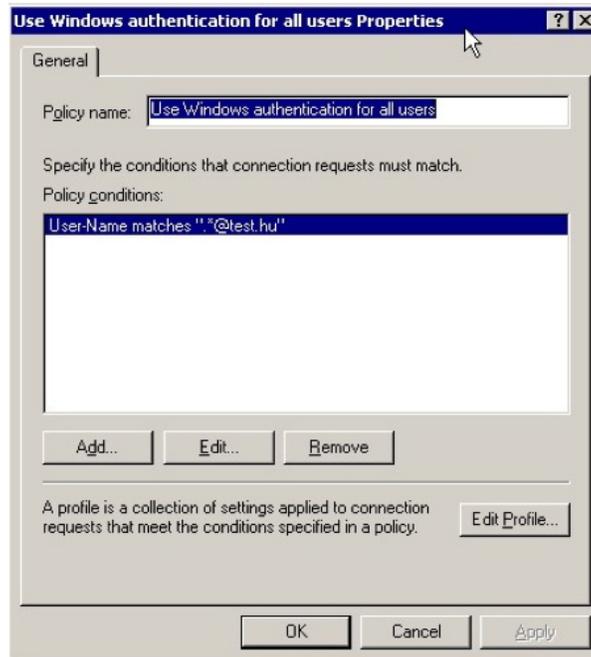
1. Configure a policy to catch local realms.
2. Configure remote RADIUS server.
3. Configure a policy that forwards all other requests to the upstream proxy server.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

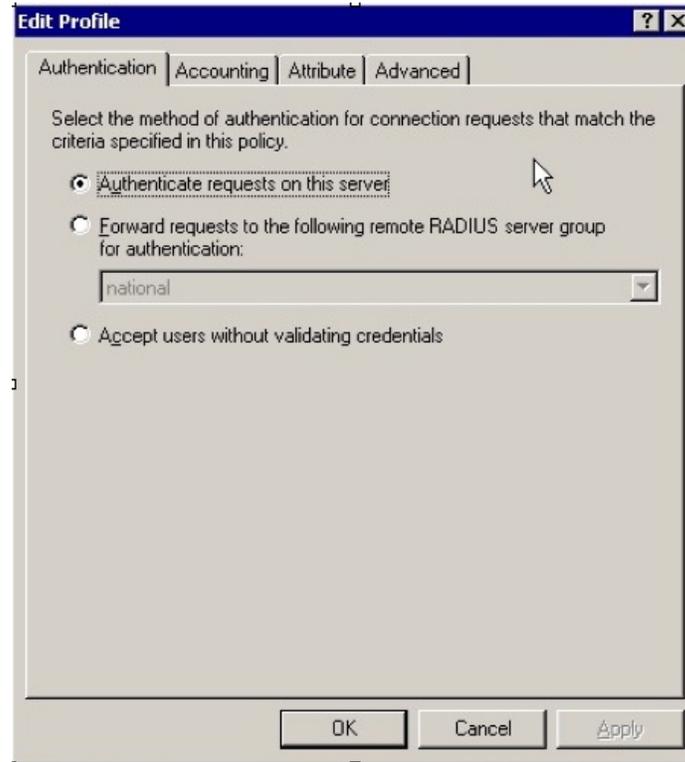
These procedures are described below.

Configuring policy for local realm

1. Configure a Connection Request Processing Policy that captures all the User-Names that are used for access to local realms using the policy condition ".*@yourrealm.tld".

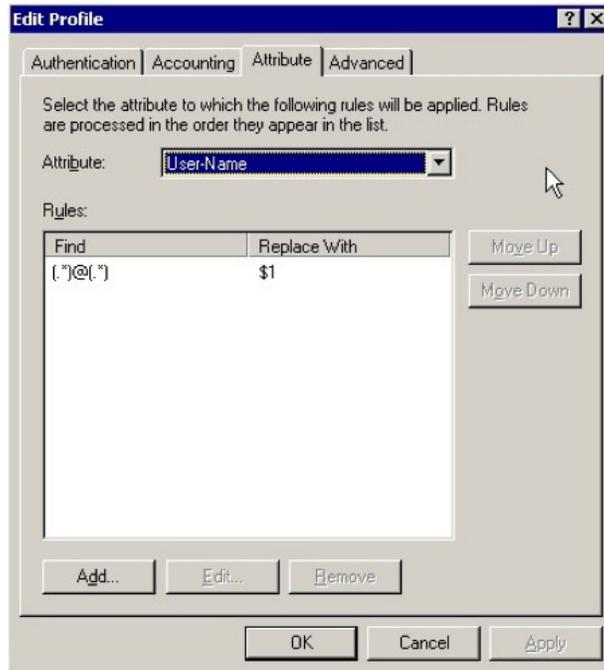


2. Click **Edit Profile** and specify that authentication occurs on the local server:



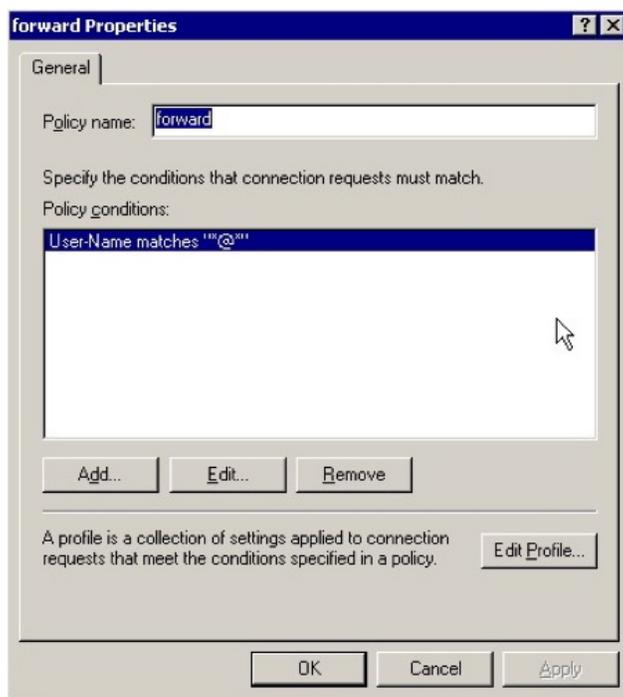
3. Click the **Attributes** tab and ensure that the RADIUS attributes are processed correctly, i.e. in the case of a matching realm name the realm name must be stripped off:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

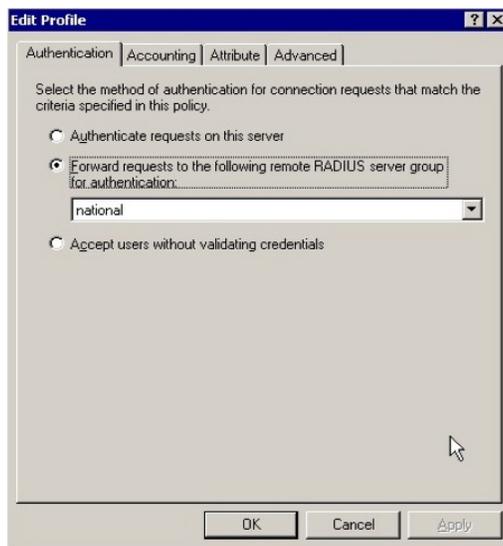


Configuring policy for upstream RADIUS proxy server

1. Configure a Connection Request Processing Policy that captures all the User-Names that are potentially used for roaming with the policy condition “.*@.*”.



2. Click **Edit Profile** and specify that authentication requests must be forwarded to the national proxy server for this profile:

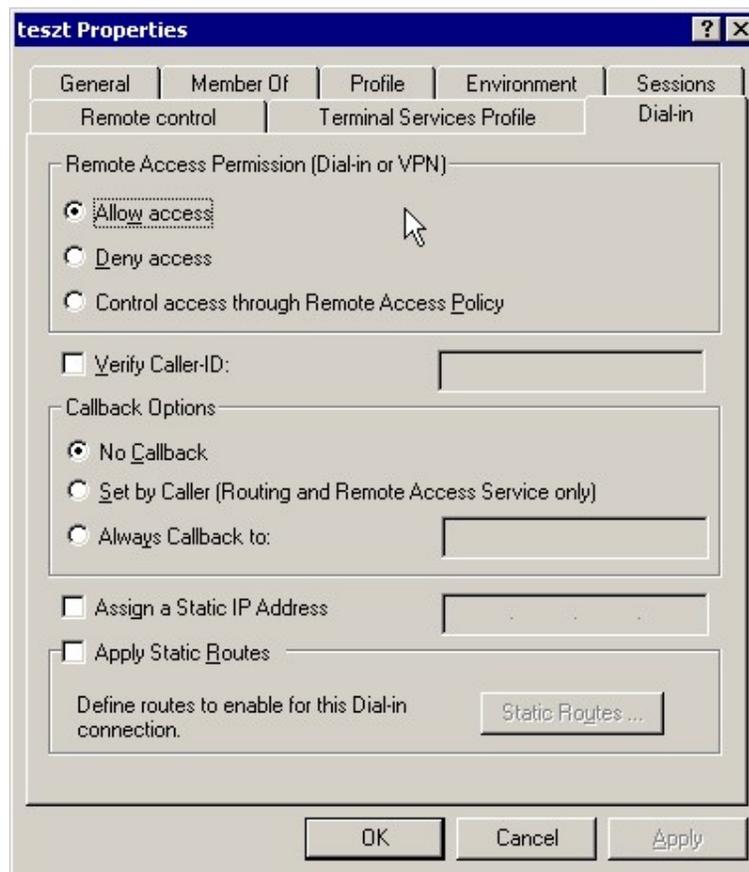


3. **Select** the remote RADIUS server group from the list.

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

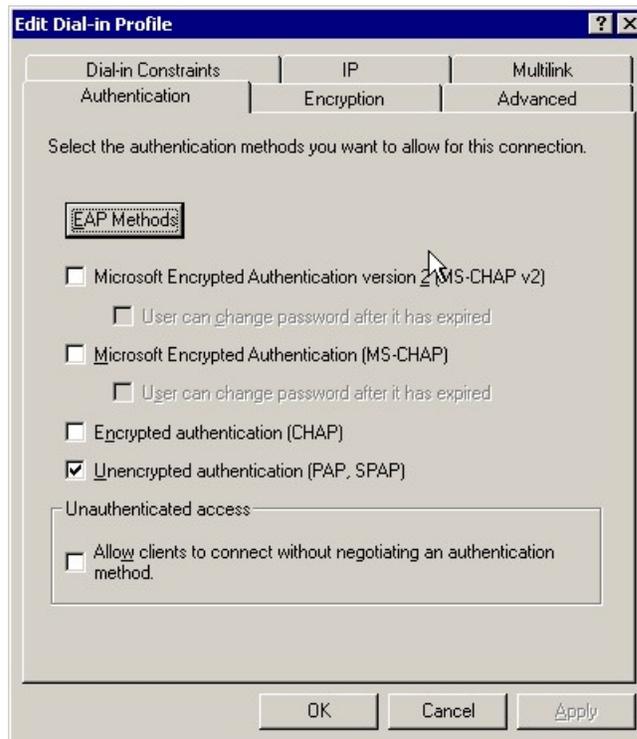
A.4.4 Configuring Domain Users to be able to use eduroam with their credentials to Windows Domain

By default, users configured in the Windows Domain are not able to use their Windows Domain username and password to authenticate against IAS. For eduroam, this should be enabled in the Domain to allow access to Remote Access Permission. This can be done using the User Management interface or the Domain Manager interface with the following policy:

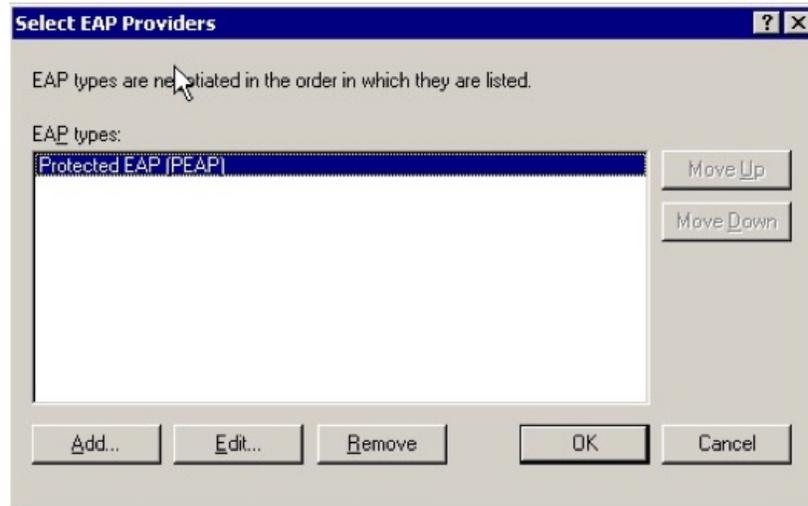


A.4.5 Configuration of Authentication methods

Authentication methods are configured in **Remote Access Policies** under the **Profile** settings. The absolute minimum that needs to be enabled is PEAP under the EAP methods, but it is useful to enable PAP as well, for debugging purpose (at least for certain accounts, e.g. for test accounts):

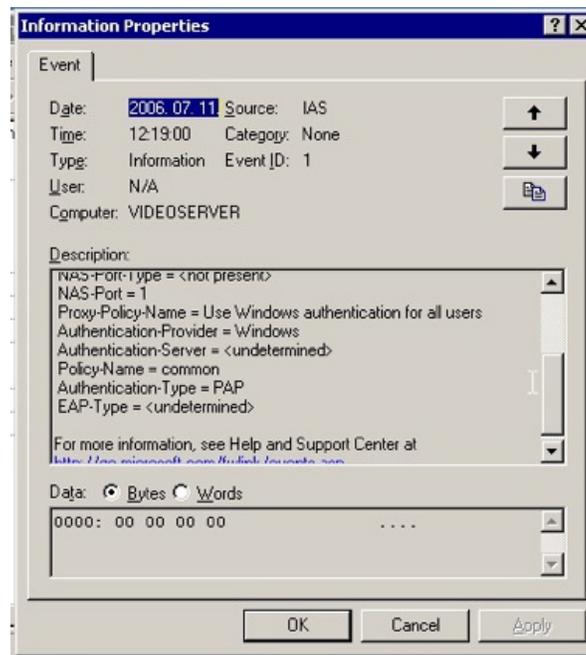


PEAP is the easiest way to deploy eduroam authentication under Windows. Deploying EAP-TLS can be labour-intensive:



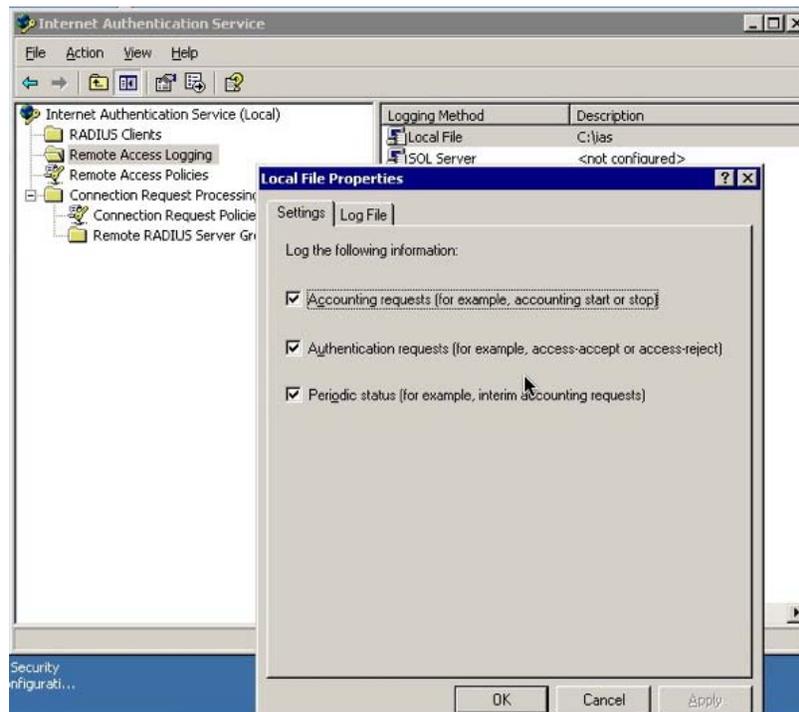
A.4.6 Troubleshooting

The most useful information can be extracted from the Eventviewer:



But you can also obtain information from the log files:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



A.4.7 References

IAS Resources: <http://technet2.microsoft.com/WindowsServer/en/Library/f6985d5d-d4c5-49e2-bbc7-385e105bfe281033.aspx?mfr=true>

Internet Authentication Service <http://technet2.microsoft.com/WindowsServer/en/Library/d98eb914-258c-4f0b-ad04-dc4db9e4ee631033.aspx?mfr=true>

IAS Pattern matching syntax: <http://technet2.microsoft.com/WindowsServer/en/Library/6e5ce48d-e662-435c-a74e-0dce305914ce1033.aspx?mfr=true>

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

Appendix B Access Points

This section describes the configuration of the Access Points:

- Cisco Aironet 1200 series.
- Lancom L-54.

These Access Points have been used extensively by JRA5 activity members, and so their configuration and suitability for eduroam are well established.

B.1 Cisco Aironet 1200 Series example setup

The required configuration can be downloaded as a file from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>. You must then amend the file as indicated by the comments in the file, and then copy and paste the commands into the Access Point in configuration mode (telnet or console access).

B.2 LANCOM L-54 Series Access Points

This series of Access Points offers a wide range of features for a mid-range price. One of the outstanding features in its price class is the ability to use ARP sniffing to determine a client's IP address even if it changes during a user session. Activating this feature fulfils the requirement for MAC to IP correlation from the confederation policy and obsoletes logging of DHCP leases.

The following steps are needed to set up eduroam on a Lancom L-54 or L-300 series access point. It describes the setup via the web interface and is current as of LCOS Version 7.52:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

B.2.1 NTP setup (confederation requirement: reliable timing source)

1. Select your Timezone under **Configuration>Date & Time>General**

General

Date and time

Time zone: +01: Berlin, Brussels, Paris, Rome

Daylight saving time: Automatic - Europe (EU)

If 'Automatic - user defined' has been selected above, here you can configure individual values for the automatic daylight-saving time changes between normal and summer time.

[Daylight saving time changes](#)

Define actions that are performed on a time basis:

[Cron table](#)

Apply Reset

2. Select **Synchronization** and check the radio button **Synchronize...**
3. Click on the link **Time Server** (NOT the menu **Time Server** on the left-hand side, which is only relevant if you want the AP to be the time server for its clients).
4. Click **Add** and enter your server details:

Time Server - Add

Name or Address: ntps1-1.uni-erlangen.de

Source IP address: INTRANET

Apply Reset

B.2.2 Logging

1. Select **Configuration>Log & Trace>Syslog** and check the box **Send information**.
2. Click **Syslog clients>add**.
3. Add (minimum) localhost: IP 127.0.0.1, and activate all sources:

IP address	Source addr.	System	Login	System time	Console login	Connections	Accounting	Administration	Router	Alert	Error	Warning	Information	Debug
<input checked="" type="checkbox"/>	127.0.0.1	On	On	On	On	On	On	On	On	On	On	On	On	On

The logs that are collected with the localhost setting will show up under **Expert Configuration>Status>TCP-IP>Syslog**.

B.2.3 Configuring the SSID

1. Select **Configuration>Wireless LAN>Logical WLAN setting – Network**.

General

This is where you can program common settings for all wireless LAN interfaces.

Country: Luxembourg

ARP handling

Broken LAN link detection disables WLAN interface

Indoor only mode

Email address for WLAN events:

Interfaces

This is where you can program further settings for each physical wireless LAN interface.

[Physical WLAN settings - Operation](#)

[Physical WLAN settings - Radio](#)

[Physical WLAN settings - Performance](#)

[Physical WLAN settings - Point-to-Point](#)

[Physical WLAN settings - Client mode](#)

[Point-to-Point partners - Point-to-Point](#)

This is where you can program further settings for each logical wireless LAN network (MultiSSID).

[Logical WLAN settings - Network](#)

[Logical WLAN settings - Transmission](#)

The following physical wireless LAN settings generally must not be changed.

[Expert WLAN settings - Beaconing](#)

[Expert WLAN settings - Roaming](#)

Apply | Reset

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

2. Click on one of the available slots, then set the following options as described:

WLAN network enabled to On.

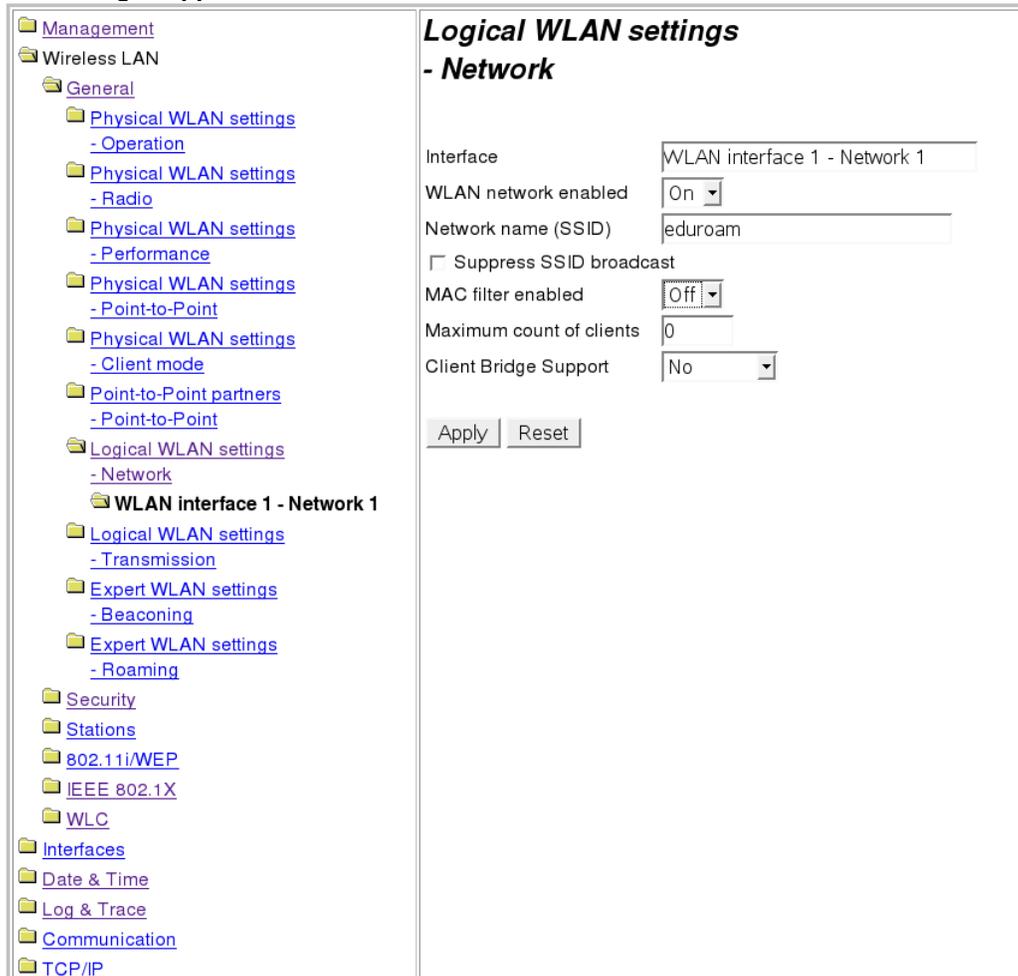
Network name (SSID) to eduroam.

Deselect the box labelled "Suppress SSID broadcast"

MAC filter enabled to Off.

Maximum count of clients to 0.

Client Bridge support to No.



Logical WLAN settings
- Network

Interface: WLAN interface 1 - Network 1

WLAN network enabled: On

Network name (SSID): eduroam

Suppress SSID broadcast

MAC filter enabled: Off

Maximum count of clients: 0

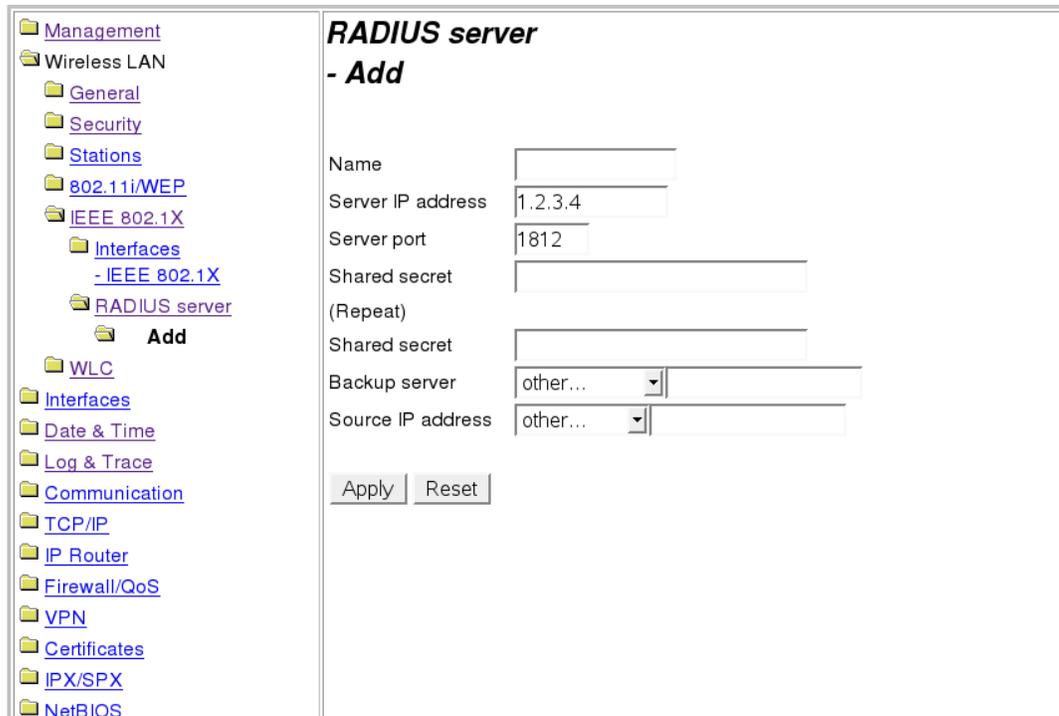
Client Bridge Support: No

Apply Reset

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

B.2.4 WPA Enterprise security

1. Configure the RADIUS server to use: Select Configuration – Wireless LAN – IEEE 802.1X – RADIUS server.
2. Click on **add** and enter your server details:



RADIUS server
- Add

Name

Server IP address

Server port

Shared secret

(Repeat)

Shared secret

Backup server

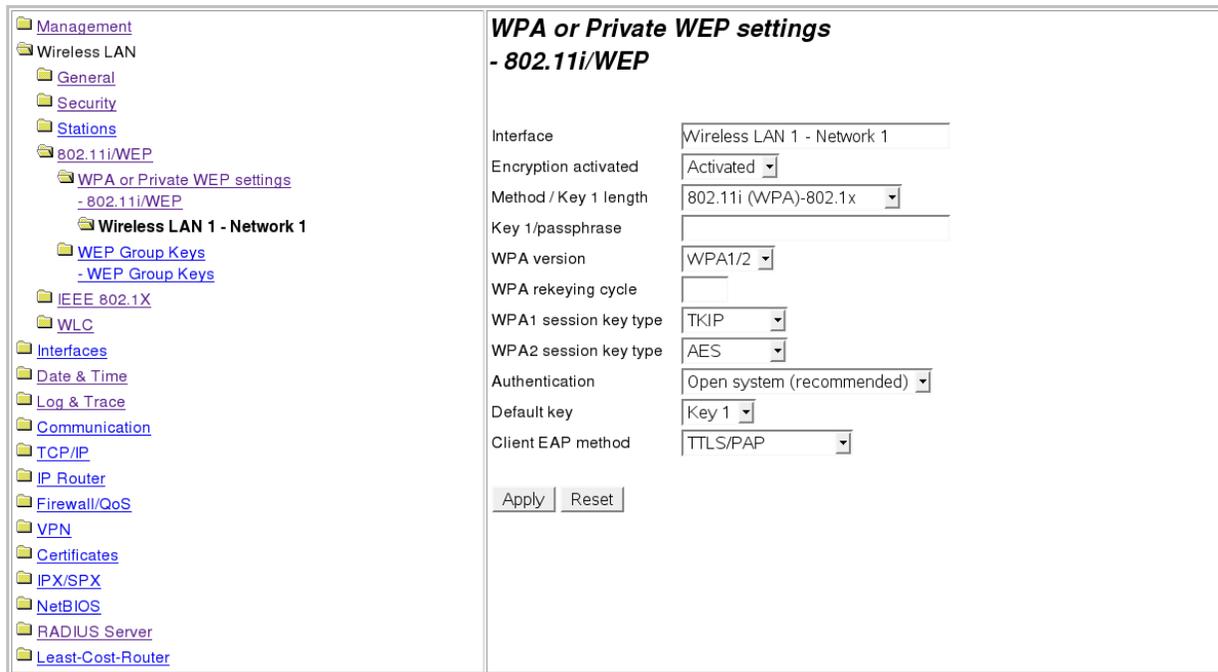
Source IP address

You must now apply the RADIUS server and encryption scheme to the SSID eduroom:

3. Select **Configuration>Wireless LAN>802.11i/WEP**.
4. Click on **WPA or Private WEP setting – 80211.i/WEP**.
5. Click on the slot in which you previously configured the SSID eduroom and enter the following settings:
Encryption Activated to Activated.
Method/Key 1 Length to 802.11i(WPA)-802.1x.
WPA Version to WPA1/2.
WPA1 Session Key Type to TKIP
 - **WPA2 Session Key Type to AES**

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

Other settings are irrelevant with WPA-Enterprise:



The screenshot shows a configuration interface for WPA or Private WEP settings. On the left is a navigation tree with categories like Management, Wireless LAN, Security, Stations, 802.11i/WEP, IEEE 802.1X, WLC, Interfaces, Date & Time, Log & Trace, Communication, TCP/IP, IP Router, Firewall/QoS, VPN, Certificates, IPX/SPX, NetBIOS, RADIUS Server, and Least-Cost-Router. The main area is titled "WPA or Private WEP settings - 802.11i/WEP" and contains the following fields:

Interface	Wireless LAN 1 - Network 1
Encryption activated	Activated
Method / Key 1 length	802.11i (WPA)-802.1x
Key 1/passphrase	
WPA version	WPA1/2
WPA rekeying cycle	
WPA1 session key type	TKIP
WPA2 session key type	AES
Authentication	Open system (recommended)
Default key	Key 1
Client EAP method	TTLS/PAP

At the bottom of the configuration area are "Apply" and "Reset" buttons.

B.2.5 RADIUS accounting server (optional)

If RADIUS accounting for the eduroam SSID shall be enabled, you must configure a RADIUS server to receive the accounting messages:

- Select **Expert Configuration>Setup – WLAN – RADIUS-Accounting** and complete the server details:
- Afterwards, activate the actual RADIUS Accounting reporting under Expert Configuration>Setup – Interfaces – WLAN – Network – RADIUS-Accounting

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

[Expert Configuration](#)

 [Setup](#)

 [WLAN](#)

RADIUS-Accounting

 Server-Address	0.0.0.0
 Acct.-Port	1813
 Secret	
 Loopback-Address	
 Protocol	RADIUS
 Backup-Server-IP-Address	0.0.0.0
 Backup-Acct.-Port	1813
 Backup-Secret	
 Backup-Loopback-Address	
 Backup-Protocol	RADIUS
 Client-Brg.-Handling	All-Traffic
 Interim-Update-Period	0
 Excluded-VLAN	0

B.2.6 Using RadSec instead of RADIUS (optional)

LANCOM devices have a RadSec client built-in. It can be used instead of standard RADIUS for the uplink to an IdP.

To use RadSec, you must have been given a eduroam Service Provider X.509 certificate from your NRO. First, upload this certificate and the eduGAIN CA certificate (which can be downloaded at <http://sca.edugain.org/cacert/eduGAINCA.pem>) via the device's "File Upload" menu:

Upload Certificate or File

Select which file you want to upload, and its name/location, then click on 'Start Upload':

File Type:

File Name/Location:

Passphrase (if required):

Caution: Files are not being checked for correct contents or passphrase during upload. These checks are performed by the individual modules using these files. When uploading certificates, possible error messages can be seen in the VPN status trace immediately after download.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

Then, go to **Expert Configuration>Setup>IEEE802.1X>RADIUS Server** and set the **Protocol** option to **RADSEC**:

[Expert Configuration](#)

[Setup](#)

[IEEE802.1X](#)

RADIUS-Server

Name	<input type="text"/>
IP-Address	<input type="text" value="0.0.0.0"/>
Port	<input type="text" value="0"/>
Secret	<input type="text"/>
(Repeat)	
Secret	<input type="text"/>
Loopback-Addr.	<input type="text"/>
Protocol	<input type="text" value="RADSEC"/>
Backup	<input type="text"/>

The same option is also present in the RADIUS Accounting server menu that was discussed above. When RadSec is to be used, we strongly suggest to use it for both authentication and accounting.

Appendix C **Supplicants**

This section describes the configuration of the following supplicants:

- SecureW2 for MS Windows.
- MacOS X Supplicant for MacOS.
- WPA_supplicant for Linux.

These supplicants are either Open Source or integrated into the Operating System, therefore there are no licence fees to consider. The main difference between these supplicants is the degree to which they can be preconfigured. Note that the WPA_supplicant can be used on either Unix or Windows platforms because of its open source nature.

C.1 **SecureW2**

In addition to the description of SecureW2 configuration in 3.2.7.1 “SecureW2”, it is also possible to use a non-preconfigured SecureW2 supplicant to connect to eduroam. This can be useful for testing purposes or for very small institutions where it does not seem worth the effort to prepare a preconfigured SecureW2.

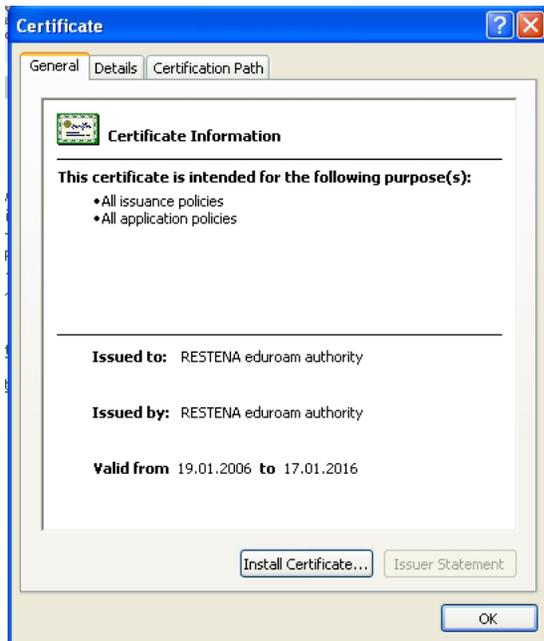
C.1.1 **Installing the CA certificate**

You will need the correct CA certificate in order to verify that you are connected to the real eduroam infrastructure. Your IdP’s configuration instructions should include an information which one that is and how to download it. If the CA certificate is a well-known one which is pre-installed on most computers, you can skip the remainder of this section. If you need to download it, please read carefully the following installation instructions.

(should be a numbered list starting at 1, but I screw this up regularly, IAT, please fix it ☺)

double-click on the certificate. The following dialog will appear:

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



Click "Install Certificate...". The following dialog box will pop up:

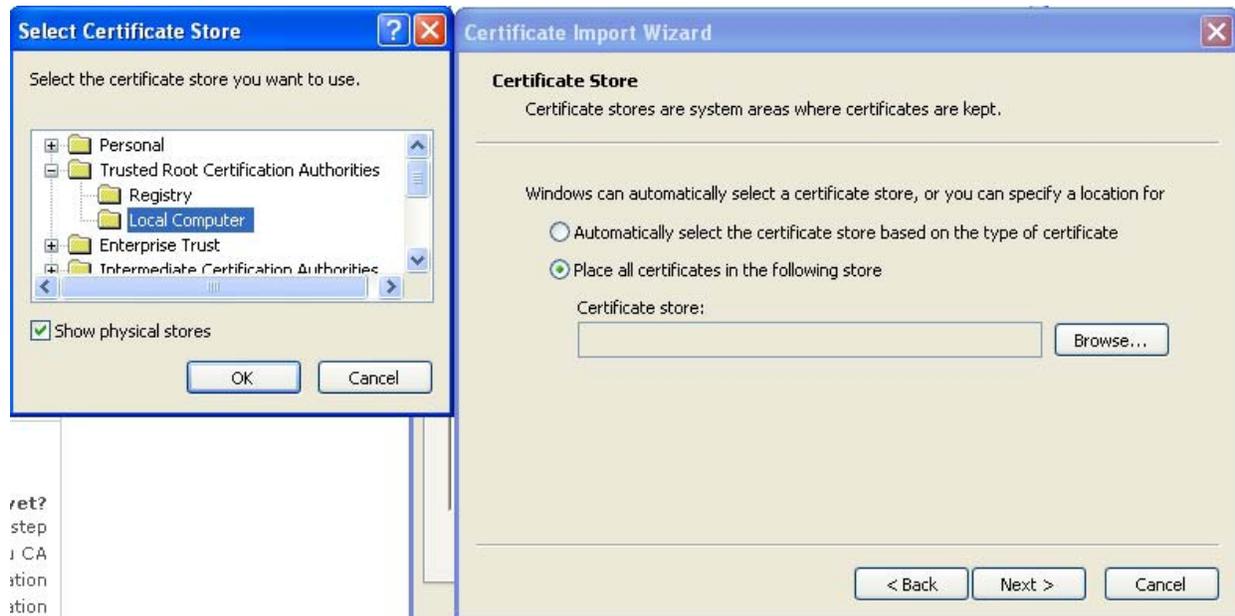


Click "Next >", then select "Place all certificates in the following store..." and push "Browse"

Check "Show physical stores"

Select "Trusted Root certification authorities", "Local computer". The result should now look like the following screenshot:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



Push “OK” and then “Next”. The certificate wizard now summarises the proposed actions:



Click “Finish”. Afterwards, the dialog box will close and you will be notified that the import was successful.

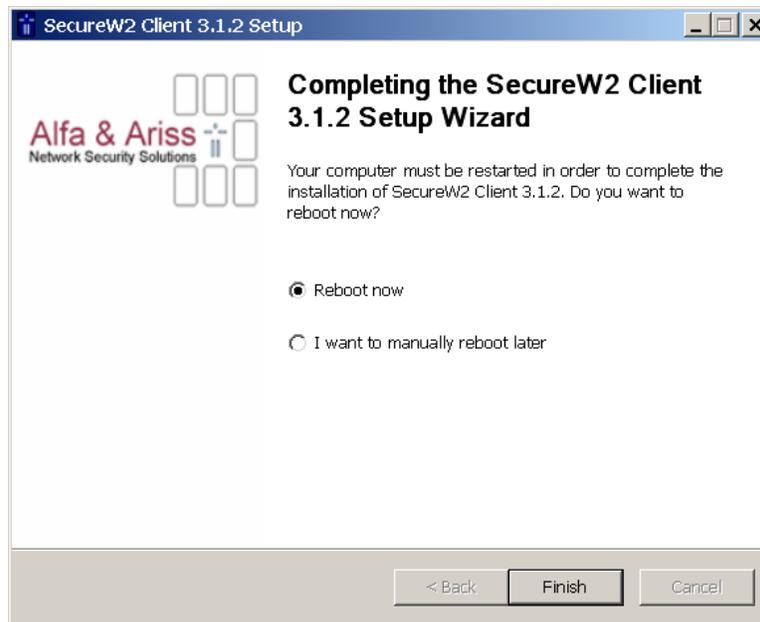
C.1.2 Installing SecureW2

To do this:

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

1. Download the SecureW2 installer from:
<http://www.securew2.com/products/>
2. Start the SecureW2 installer.
3. Click **Next**, and accept the EULA.
4. Click **Install** in the options panel.

After successful installation the following display appears asking you to reboot:



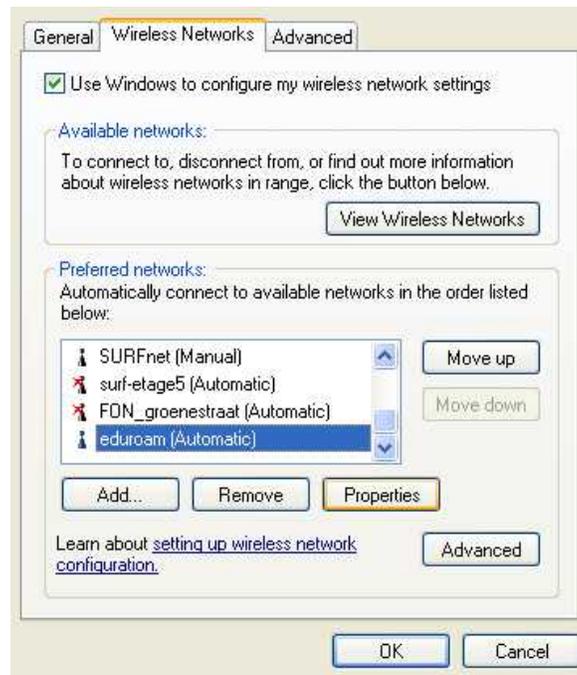
Reboot as required.

C.1.3 Configuring SecureW2

After reboot, the WLAN connection must be configured:

5. Go to the network properties and right-click on **Wireless Networks** (where you also choose properties).
6. Select **Wireless Networks** and click **Add**:

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



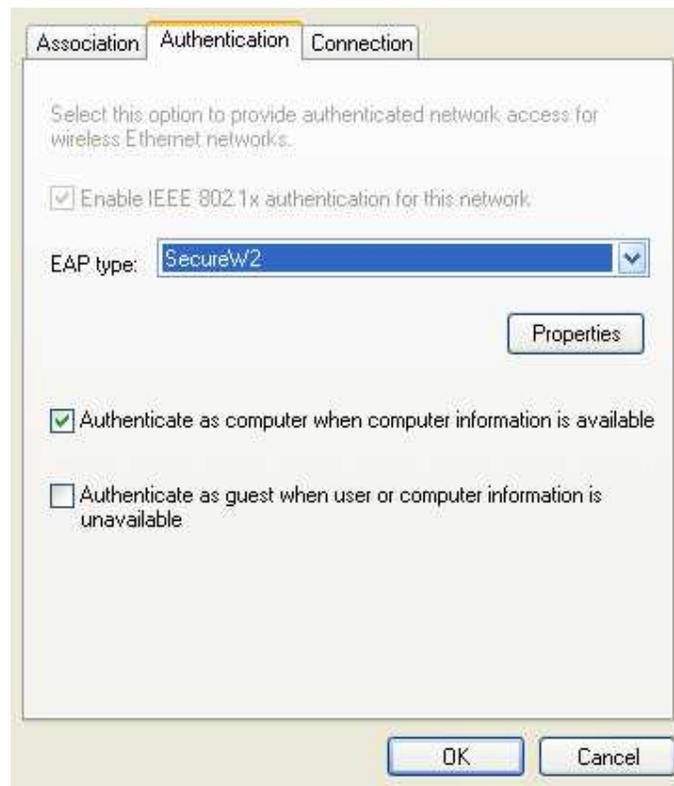
7. Enter the SSID **eduroam** and select WPA2/AES or, if your adapter does not support this, **WPA/TKIP**, as encryption.

Note: If you cannot select this, it is possible that the firmware of your WLAN adapter might need to be updated, and/or you need to install the WPA patch for XP SP2. (Knowledge Base Article 917021). That patch is not necessary for Windows XP SP3.

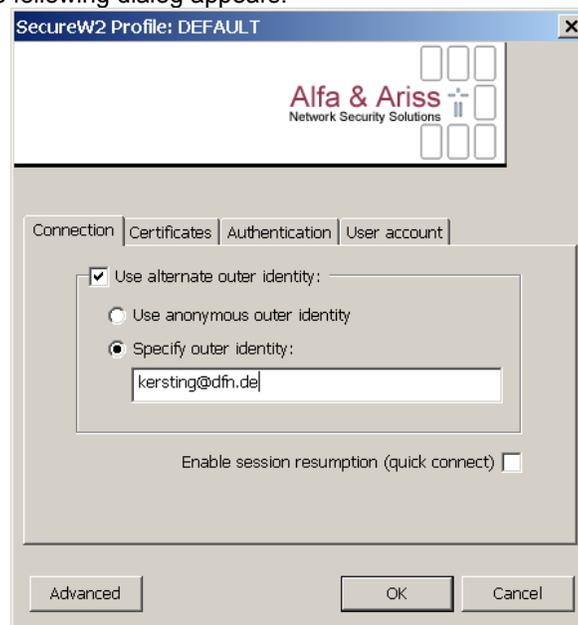
Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



8. Click the **Authentication** tab and select **SecureW2** as **EAP-Type**:



- 9. Click **Properties**.
- 10. Click **Configure**. The following dialog appears:

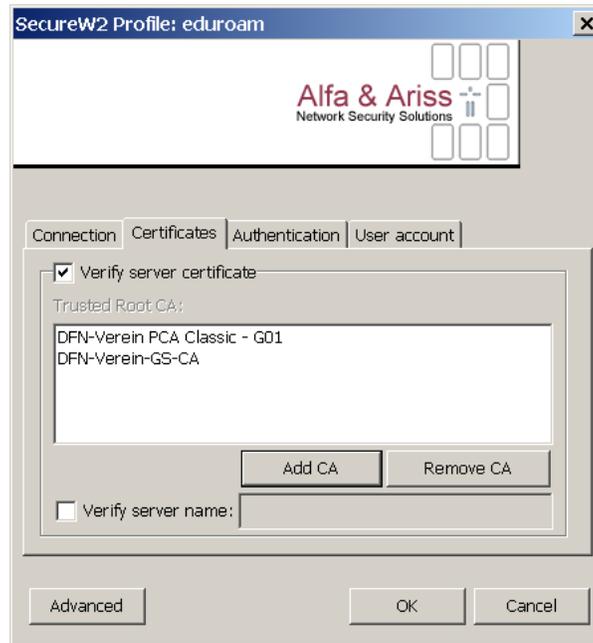


- 11. Select a meaningful profile name (preferably **eduroam**) and set your outer identity in the form

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

@yourrealm (nothing left of the @ sign).

12. Check the **Verify server certificate** checkbox and add the certificates from your CA. You will need to import the CA into the “Local Machine” Store for “Trusted Root Certification Authorities” first, otherwise it will not show up in this list (see .
13. Check the **Verify server name** and add the DNS name of your RADIUS server:



14. Click the **Authentication** tab and verify that **Authentication Method** is set to **pap**
15. Click the **User account** tab:

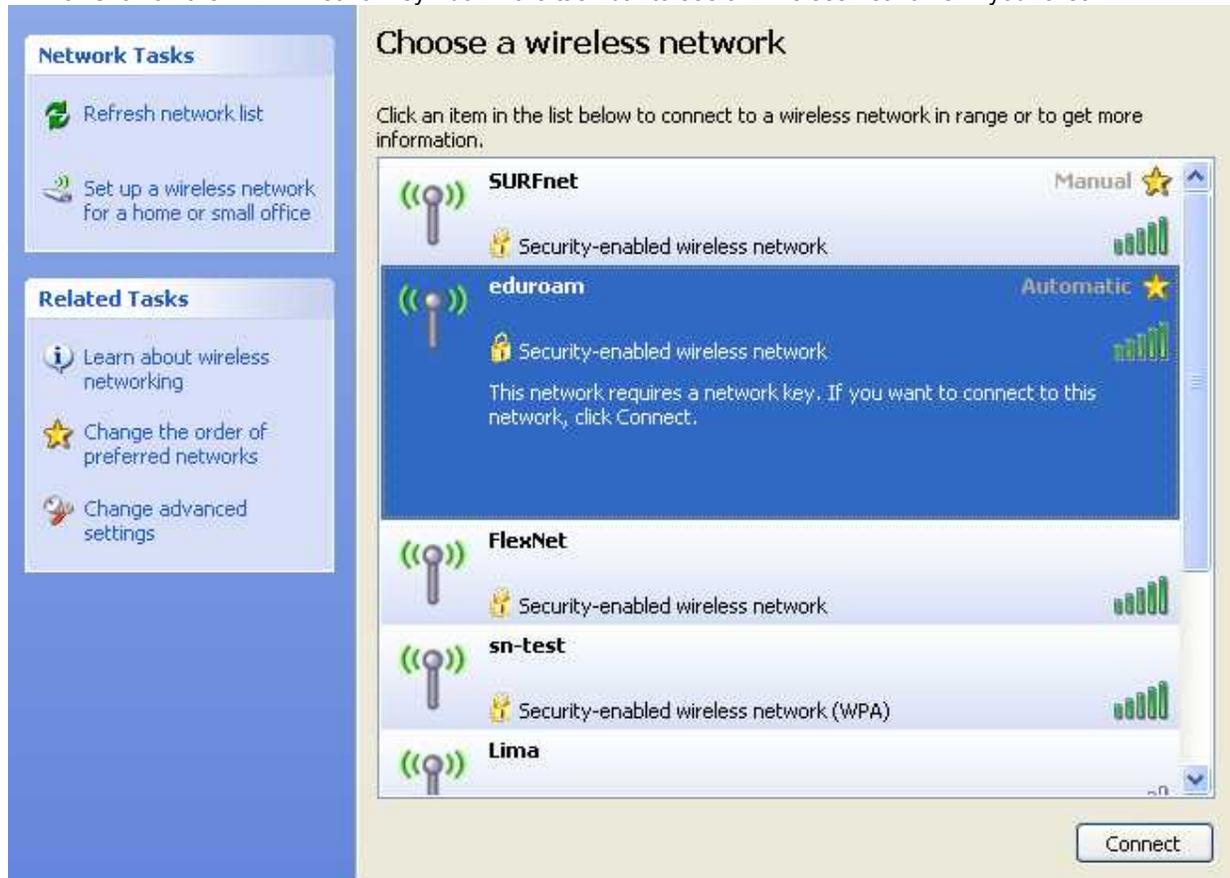


16. Enter your username (in the format **yourname@yourrealm**) and password and click **OK**.
17. Close all open windows. You are now ready to connect.

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

C.1.4 Connecting to eduroam

18. Click on the WLAN network symbol in the task bar to see all wireless networks in your area:



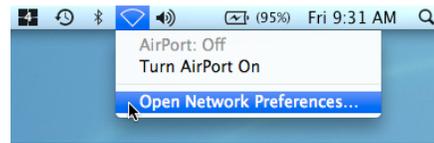
19. Select **eduroam** and click **Connect**.

C.2 MacOS

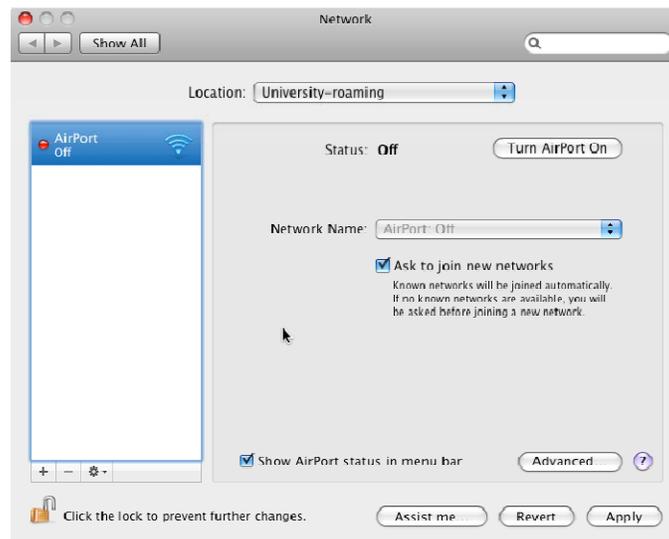
These instructions are valid for Mac OS X 10.5 (Leopard) update 4. Please note that previous updates of Leopard had a different user interface. Future updates could also change this interface and the instructions in this document could be wrong or not accurate.

1. Open **Network Preferences**. This can be accessed either from the Systems Preferences panel or the menu bar:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



2. If required, create a new **Location** for roaming purposes. You can create different locations for ease of use (such as **Home** or **Work**). In this example, the Location **University-roaming** is used. Just the wireless interface (also known as AirPort in Mac OS X) exists for this location:



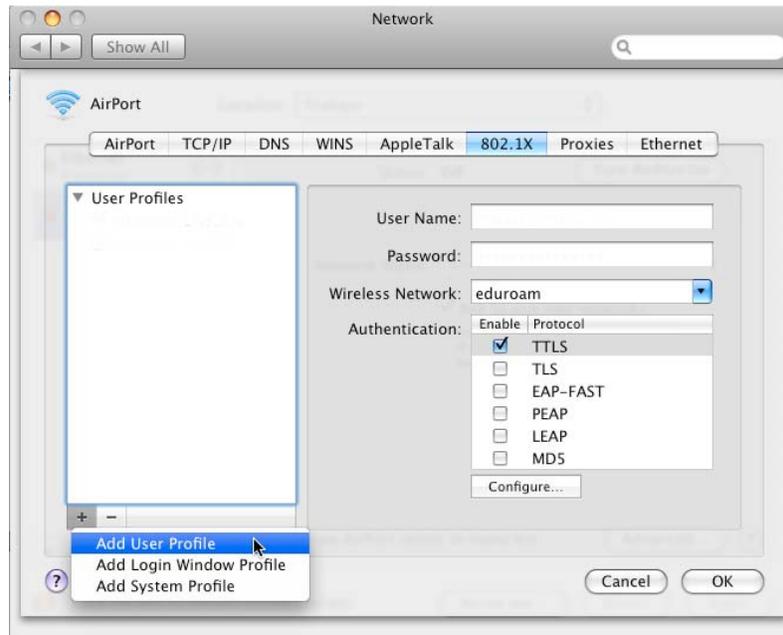
3. Click on **Advanced** to open the **Advanced Options** panel for the AirPort interface. Of the tabs available on this panel, the following will be used to configure eduroam:

- 802.1X. This tab is for configuring 802.1X access profiles. Please note that these profiles can be used for several Locations.
 - o *AirPort*. This tab will display when the interface is **On** or a network has been manually configured. It displays the identifier for the different networks and security being used when connecting to them (note that several security modes with the same identifier could be configured, for instance for WPA

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

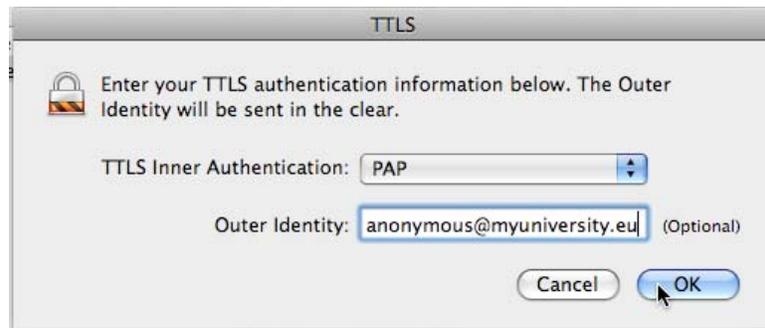
and WPA2).

4. Click on **802.1X** to create an 802.1X User Profile. The 802.1x tab screen appears.
5. In the Profiles panel, click the plus (+) symbol and select **Add User Profile**:

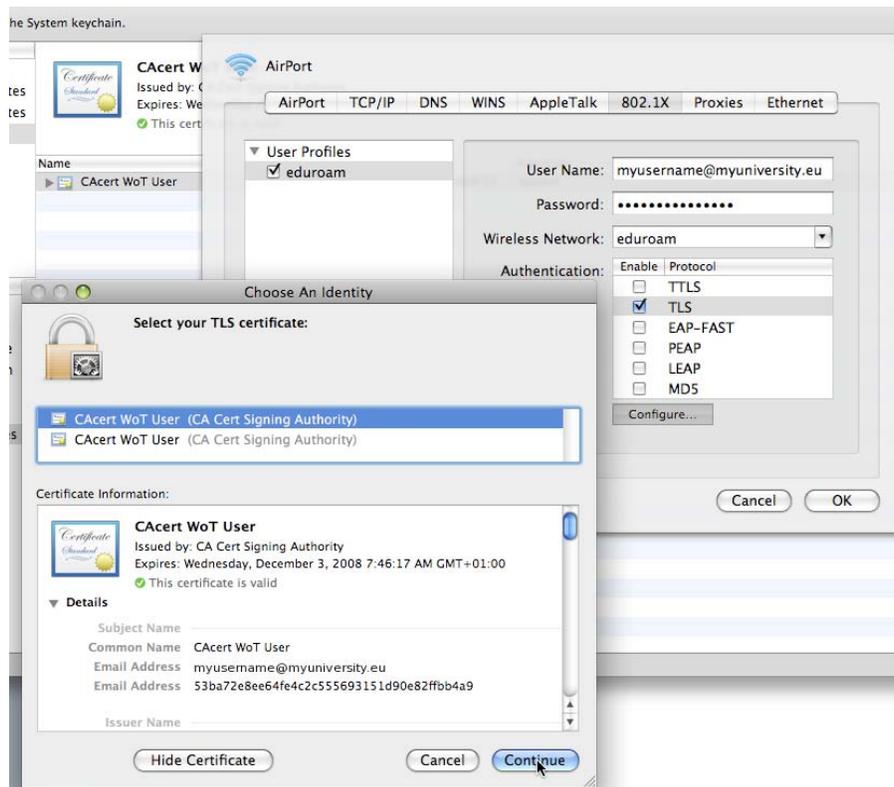


Note: The commonest requirement is to create a user profile. However, you can also create a profile for authenticating users to the system, or a common profile for all the users.

6. A User profile example dialogue appears. The minimum information you must enter for a user profile is: Username and password (the actual username and password of the user).
The default wireless network identifier to join using this profile.
The authentication method to be used. This can be one of (most commonly):
 - TTLS, which supports four inner authentication methods: MSCHAP, MSCHAPv2, CHAP, and PAP. You can also provide an inner identity (optional, but recommended for eduroam). The example below uses the prefix *anonymous*. However, just the postfix with the @ symbol and the user home institution realm is acceptable.

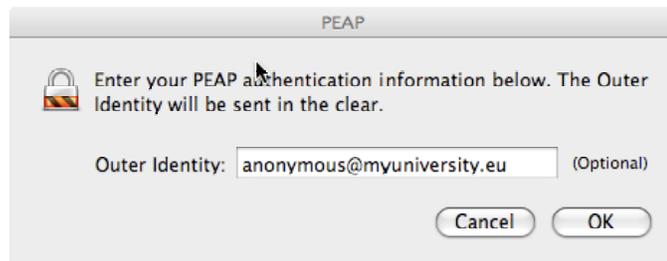


- TLS. This method allows us to authenticate using a user certificate. The certificate must be installed in the user's keychain (via the KeyChain Access application) for this option to be activated. After activating the check box, a dialogue for selecting the appropriate certificate is shown. In this example we use a certificate signed by the CA Cert Signing Authority:

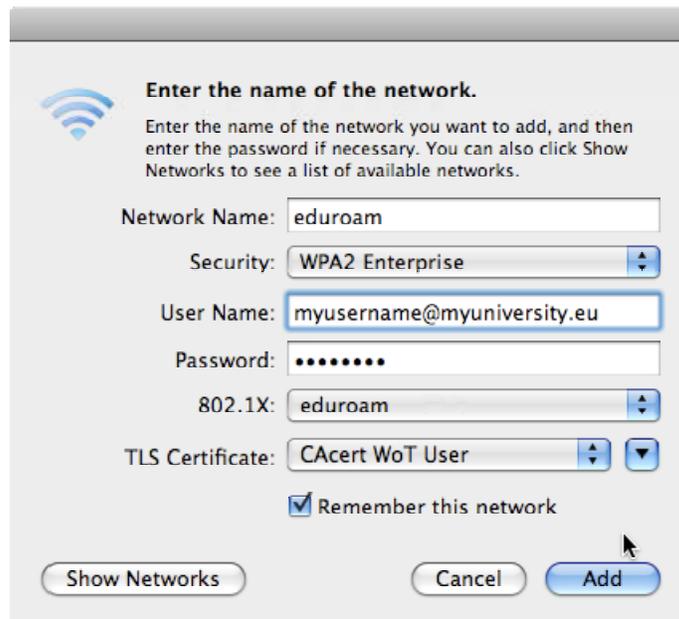


- PEAP. This authentication method also offers the possibility for entering the outer identity:

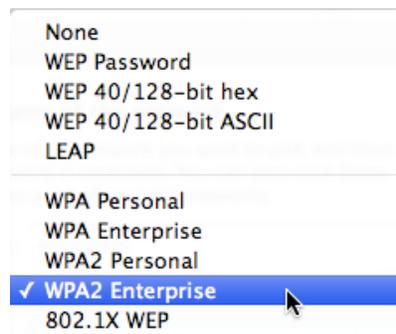
Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



7. Select the **AirPort** tab and use the plus (+) icon to add the eduroam network using the previously created 802.1x profile. The following dialogue appears:



8. Click on the **Security** box to display the following:



9. In most cases select either **WPA Enterprise** or **WPA2 Enterprise**.

Note: By selecting the 802.1x profile created in previously (field 802.1X), fields containing user

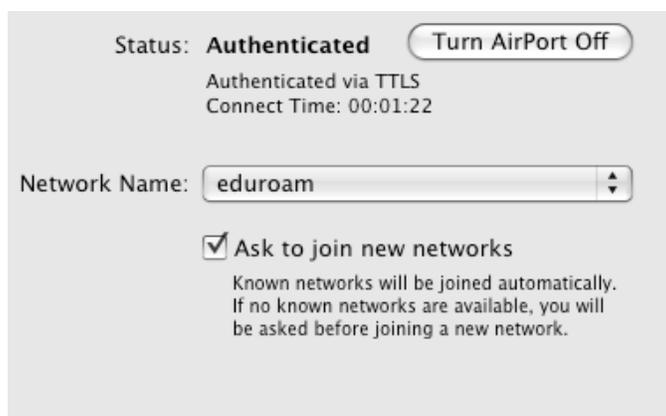
Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

credentials are automatically populated with the values previously entered.

10. Activate the **Airport** interface.

11. Apply the changes made in the previous steps and activate the network, either from the **Network Preferences** panel or the menu bar.

Assuming the configuration was successful, you will see something like the following::



While connecting to the server, it's possible that a certificate warning could appear. This happens when:

- The certificate presented by the RADIUS server being contacted is valid but not signed by any of the Certification Authorities recognised by your computer. To prevent this, any new Certification Authority should be added using the KeyChain Access tool. If you are confident that you are talking to the right server, and that the certificate presented by the server contains all the certification path, you **can** trust this certificate (any additional checks can be carried out by expanding the certificate information if necessary).
- The certificate presented by the RADIUS server is invalid or self-signed. In this case **do not trust** the certificate.

C.3 iPhone

The **iPhone Configuration Utility** allows you to set up your iPhone to access eduroam. This tool helps you create a wireless configuration profile (.mobileprofile file extension). There are versions for Windows and Mac OS X.

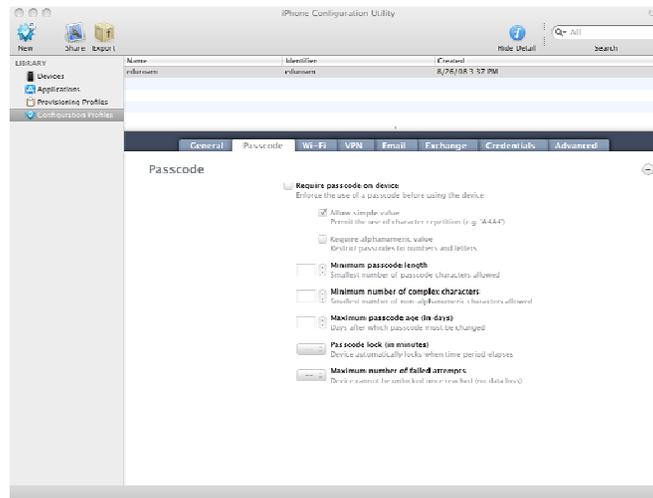
You can also download profiles sent to your iPhone.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

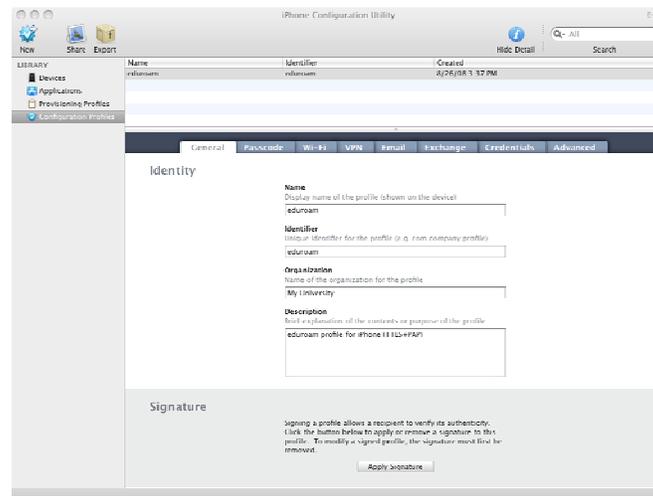
C.3.1 Using the configuration utility

To configure your iPhone for eduroam:

1. Start the iPhone Configuration utility.
2. Select the **Passcode** tab and select **Allow simple value**:

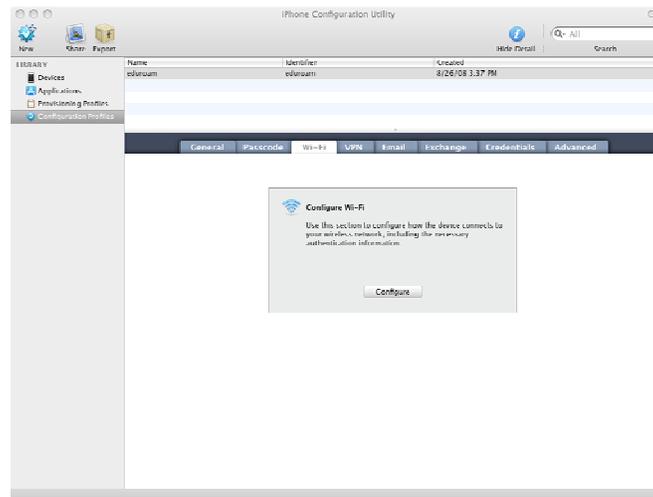


3. Select the **General** tab and enter **Identity** details for this profile:
 - Name:** Enter **eduroam**.
 - Identifier:** Enter **eduroam**.
 - Organisation:** Enter the name of your organisation.
 - o **Description:** Enter a description for this profile.

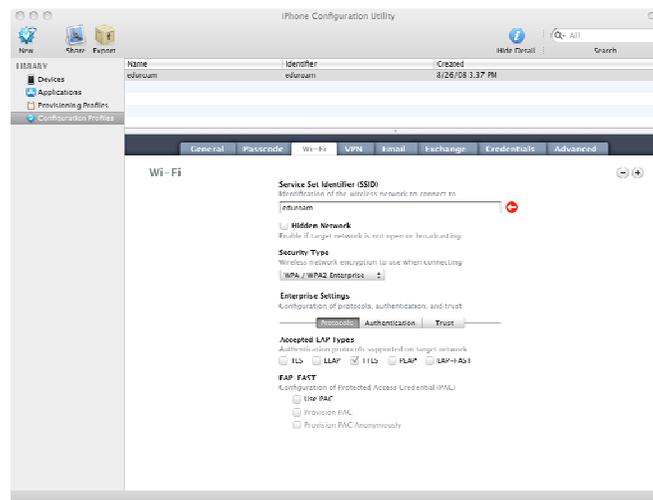


4. Click the **Wi-Fi** tab

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

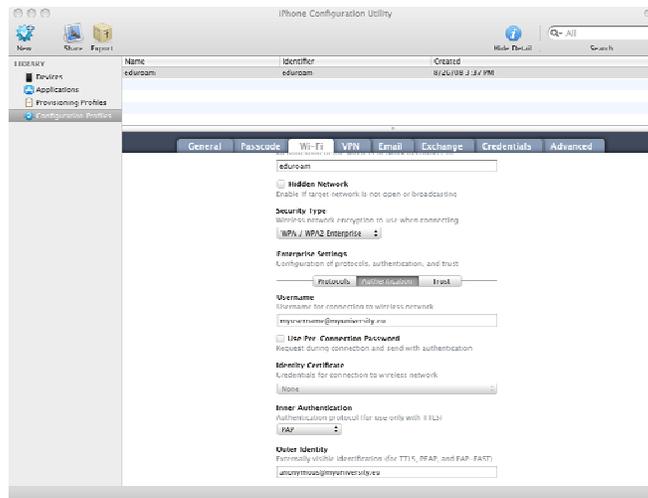


5. Click **Configure** to enter connection and authorisation information as follows:
Service Set Identifier (SSID): Enter **eduroam**.
 - Security Type: Select WPA/WPA2 Enterprise.
6. Under **Enterprise Settings** click **Protocols** and under **Accepted EAP Types** select **TLS**:

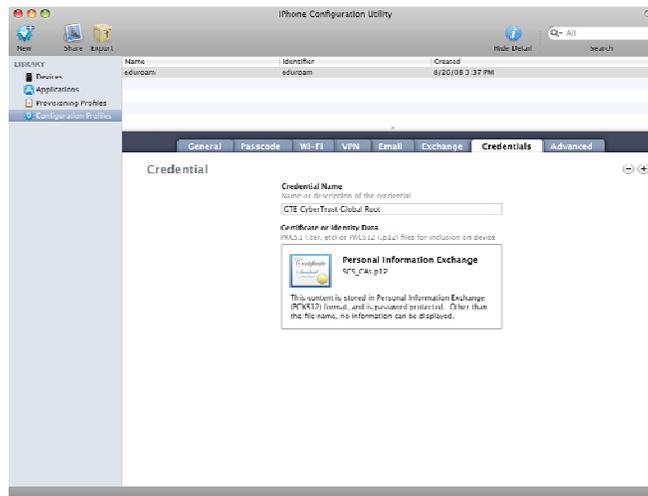


7. Under **Enterprise Settings** click **Authentication** and enter authentication details:
Username: Enter your email address.
Inner Authentication: Select **PAP**.
Outer Identity: Enter the identity to be used for external identification (e.g. anonymous@myuniversity.eu).

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



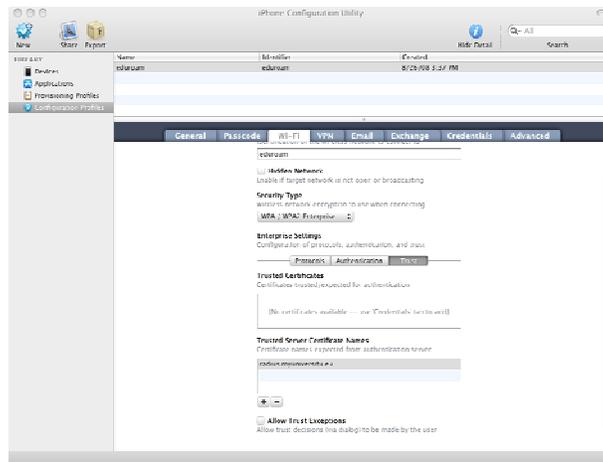
8. Click the **Credentials** tab and enter the **Credential Name** to be used.



9. Click the **Wi-Fi** tab and under **Enterprise Settings** click **Trust** to enter trust details:

- **Trusted Server Certificate Names:** Enter the names expected from the Authentication server.

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



C.3.2 Downloading a profile

You can receive a profile on your iPhone. An example is shown below:



Opening the attachment to the message displays a dialogue such as:

Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230



Note: In this example the profile has not been signed. However the profile contains a valid TLS+PAP configuration and the right certificates for recognising the organization radius server certificate.

If you decide to install this profile, you will be connected to eduroam:



Project:	GN2
Deliverable Number:	DJ5.1.5.3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

C.4 WPA_Supplicant

Wpa_supplicant (http://hostap.epitest.fi/wpa_supplicant/) is an open source 802.1x supplicant for Linux, BSD and MS Windows. This section describes its use on the Linux platform. Wpa_supplicant is available for most modern Linux distributions and seems to be the focal point of 802.1X development on the *nix platform.

It is out of scope of this cookbook to describe how wpa_supplicant can be compiled from source or what options need to be enabled in the Linux kernel to make eduroam authentication work. Modern Linux distributions with standard kernel, wireless tools and wpa_supplicant should work “out of the box”.

Using the technical information below it is possible to implement eduroam support so that it will be seamlessly integrated with the OS. This is, however, very distribution specific and therefore out-of-scope for this document.

It is assumed that the user has a working wireless card (this can be verified by using the `iwconfig` command).

wpa_supplicant is responsible for the (layer 2) authentication of the user, and must be followed by some means of setting up the (layer 3) IP connection by using a DHCP-client. Wpa_supplicant typically runs in the background to control the connection, take care of re-authentications, manage roaming between access points, and so on. It is started with the command:

```
wpa_supplicant -i interface -c configuration_file -D driver -B
```

Where:

- **interface** is the system name for the wireless interface (like eth1, ath0, wlan0, etc.).
- **configuration_file** is the location of the file (described later on).
- **driver** is one of: wext, ipw, madwifi and ndiswrapper (described below).
- **-B** option means ‘run in the background’.

The driver setting depends on the particular card used.

The wext driver currently supports most existing cards (Atheros chipset based cards being an exception, madwifi should be used there). Hence, the wext setting should be tested first.

The configuration file depends on the EAP type of choice. Example configurations (which can be downloaded from <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>) are provided for EAP-TTLS, EAP-PEAP and EAP-TLS. Each of the examples contains two, nearly identical, network blocks; the only difference is that one is for WPA and the second for dynamic WEP.

Note: In principle, one block with ‘key_mgmt=WPA-EAP IEEE8021X’ should be sufficient, but under certain conditions this may fail. Using two separate blocks always seems to work correctly.

Also, The `ca_cert` points to the certificate file of the CA which has provided the certificate for the RADIUS server. This file should contain certificates for the whole certification chain, up to the root. All certificates and

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

keys should be in PEM format.

The EAP-TTLS file is shown below for reference:

```
# EAP-TTLS configuration
ctrl_interface=/var/run/wpa_supplicant

network={
    ssid="eduroam"
    key_mgmt=WPA-EAP
    ca_cert="/etc/eduroam/ca.cer"
    identity="user@your.domain"
    eap=TTLS
    password="xxxx"
    phase2="auth=PAP"
}

network={
    ssid="eduroam"
    key_mgmt=IEEE8021X
    ca_cert="/etc/eduroam/ca.cer"
    identity="user@your.domain"
    eap=TTLS
    password="xxxx"
    phase2="auth=PAP"
}
```

Note: this example will set the outer identity to be the same as the real, inner identity of the user. It is possible to set the outer identity to a different name (for instance to an opaque id), but for simplicity this is not shown here.

Also, most wpa_supplicant compilations will accept user key/certificate in one PFX (p12) file. If that is used, this file should be pointed to by `private_key` and `client_cert` should be commented out.

Download the script (contained in <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>) to enable you to start and stop the eduroam connection. Note that this script needs to be configured by assigning correct values to the variables in the configuration section. The script kills possible wpa_supplicant processes and DHCP clients for the particular interface. Then it starts wpa_supplicant and monitors its state with `wpa_cli`. If no authentication takes place during the `REAUTH_TIMEOUT` period, wpa_supplicant is restarted. After authentication, the DHCP client is started.

Note: This script has to be run with administrator's rights. This can be avoided by creating wrappers, which can then be connected to panel buttons so the network can be started and stopped by a mouse click and providing the administrator's password. To enable this, the following package can be downloaded and used:

http://eduroam.pl/Files/prepare_eduroam_config.tgz.

This utility allows campus administrators to create a configuration script that can be distributed to the users. The script contains all necessary certificates, scans the system for the needed tools and creates configuration files, certificate files, sets up the main starting script and wrappers. A full description is beyond the scope of this document, but can be found in the documentation of the package.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

C.5 Intel PROSet/Wireless supplicant

This supplicant (shipped with Intel Centrino chipsets) makes it very easy for an administrator to prepare and distribute a pre-configured supplicant configuration.

C.5.1 Preparing the profile as an administrator

As an IdP administrator, first create a configuration for yourself in the supplicant. It is suggested that you use a proper anonymous outer identity of “@realm” (your realm, with nothing left of the @ sign) as the “Roaming Identity”. Now, verify that the settings are correct by connecting to eduroam.

Then, in the main window of the supplicant, click on “Profiles...”. In the Profiles window, select the newly created eduroam entry and then press the “Export” button. You will be prompted for the destination folder of the profile. The filename in this folder will be the name of the profile as it is shown in the Profiles window.

Note that the exported profile will be exported without your username and password, but with the anonymous identity preserved, i.e. your own credentials are automatically safe from inadvertent leakage.

Distribute the generated file to your users.

C.5.2 Installing the profile as a user

A simple double-click will install the profile on the user’s supplicant. When the user tries to connect for the first time, he will be prompted for his own username and password.

Note that there will be no visual feedback at all after double-clicking on the profile file. A user might be tempted to think that nothing happened and try to import the profile twice. In this case, an error message about the duplicate profile will be displayed. We suggest to educate the users accordingly.

Appendix D **eduroam along with commercial hotspot system**

This chapter describes a sophisticated deployment of Wireless LAN that includes both eduroam access (as service provider, not identity provider) and a commercial hotspot deployment that offers three distinct classes of access and multiple billing models. The following access models are covered:

- Commercial web redirect login portal.
- Commercial WPA2-Enterprise secured access.
- WPA2-Enterprise access for institution staff (not eduroam-eligible).
- eduroam access for R&E users.

Every usage class is assigned a separate VLAN and allows isolation of users. Four SSIDs are in use, and are named “ccrn-hotspot” (web redirect), “ccrn-wpa” (commercial WPA), a hidden SSID (staff access) and “eduroam”.

The billing models for the commercial access allow:

- Online-time based billing.
- Time window based billing.
- Volume based billing.

This mixture of a commercial system and eduroam access can be applied to any operator who is not in the Research and Education community. The commercial system generates revenue while offering eduroam is a competitive advantage against other providers. Several deployments of eduroam in non-R&E locations have shown that eduroam attracts students to these places and by that may generate extra revenue – pubs are a prime example for this business model.

The following instructions demonstrate how to set up such a hotspot solution with a single hardware server, switches, Access Points and pure Open Source Software. This is a real-life scenario deployed in the city of Luxembourg at the “Centre Culturel de Rencontre Abbaye de Neumünster” (CCRN).

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

these instructions assumes that a server with at least two network interfaces is present, where eth0 connects to the outside internet, and eth1 is free for use with the hotspot system. It uses IP addresses in the 10.10.0.0/8 range within the system. The instructions attempt to be distribution-neutral. However, users should note that this example was installed on an openSUSE 10.2 Linux operating system, and distribution-specific information may be present

D.1 Installation Instructions

1. Prepare a Linux server with a distribution of choice and install the following packages at a minimum:
 - vconfig -> provides the VLAN configuration tool vconfig (separate download required: <http://www.candelatech.com/~greear/vlan.html>).
 - chillispot -> provides the web-redirect portal binary, chilli (version 1.1.0 is on openSUSE 10.2 installation media).
 - iptables -> provides firewall manipulation tools iptables, ip6tables (version 1.3.6 is on openSUSE 10.2 installation media).
 - apache2 -> provides the web server for the web-redirect portal httpd (version 2.2.3 is on openSUSE 10.2 installation media).
 - MySQL -> provides the datastore for user accounts mysql (version 5.0.26 is on openSUSE 10.2 installation media).
 - apache2-mod-perl -> enables execution of perl CGIs (version 2.0.2 is on openSUSE 10.2 installation media).
 - php5 -> provides php (version 5.2.0 is on openSUSE 10.2 installation media).
 - phpmyprepaid -> provides user management web interface (separate download required: <http://sourceforge.net/projects/phpmyprepaid>, in this deployment version 0.3.3 is in use).
 - freeradius -> provides the RADIUS server radiusd (version 1.1.3 is on openSUSE 10.2 installation media).
 - o dhcp-server -> provides the DHCP server dhcpd (version 3.0.5 is on openSUSE 10.2 installation media).

2. Ensure the following configurations are met:

Kernel: must support
IEEE 802.1q VLANs
tun/tap network interfaces
netfilter
must have routing capabilities

Note: The openSUSE 10.2 kernel supports all of the above.

VLANs: add with

```
vconfig add eth1 10  
vconfig add eth1 11  
vconfig add eth1 12  
vconfig add eth1 13
```

assign IP addresses to

```
eth1:      ifconfig eth1 10.10.0.1 netmask 255.255.255.0 up
eth1.10:   ifconfig eth1.10 10.10.10.1 netmask 255.255.255.0 up
eth1.11:   ifconfig eth1.11 10.10.11.1 netmask 255.255.255.0 up
eth1.12:   ifconfig eth1.12 10.10.12.1 netmask 255.255.255.0 up
```

Do NOT assign an IP address to eth1.13 but make sure the interface is running (ifconfig eth1.13 up)

Make sure routing is turned on (cat /proc/sys/net/ipv4/ip_forward must give the result "1").

chillispot:

INTIF is eth1.13

EXTIF is eth0

set uamsecret to an arbitrary value; the same value must be in CGI configuration below

RADIUS server is localhost; use the shared secret for localhost (typically testing123)

copy hotspotlogin.cgi into apache's cgi-bin store

copy dictionary.chillispot into /etc/raddb

iptables:

modify ruleset to allow RADIUS traffic in INPUT chain from eth1

modify ruleset to allow DHCP requests in INPUT from eth1.10, eth1.11, eth1.12

apply ruleset

MySQL: create database "radius" and user for access

Apache2:

make sure CGI support for perl is active

make sure PHP support is enabled

request a certificate from a well-known authority that includes the TLS Web Server Authentication

OID (for example, Thawte and Verisign include this OID)

SSL support on port TCP/443 is mandatory: install the server cert

phpmy prepaid:

install somewhere in apache2 document root

set database details in dbconnect.php

point browser at installation path and follow instructions

set up a "location" first, then billing models

It is not necessary to define Access Points

dhcpd

define DHCP ranges for subnets 10.10.10.1/24, 10.10.11.1/24, 10.10.12.1/24

don't bind on interfaces eth0, eth1 and eth1.13

FreeRADIUS:

use database "radius" on localhost as authentication source

\$INCLUDE dictionary.chillispot into file dictionary

define a realm DEFAULT that points to the eduroam infrastructure

realms NULL and LOCAL have auth source LOCAL

tag incoming requests in "hints" file to match SSIDs and user auth sources

allow EAP passthrough for eduroam users

— install server cert for EAP sessions for own users (-wpa, -staff)

It is useful to put VLAN definitions, IP allocations, firewall ruleset application into an init script to automate the boot process, an example init script is provided at <http://www.eduroam.org/downloads/docs/eduroam-cookbook-scripts.zip>.

Project:	GN2
Deliverable Number:	DJ5.1.5,3
Date of Issue:	29/10/08
EC Contract No.:	511082
Document Code:	GN2-08-230

3. Add dhcpd, mysql, apache2, freeradius, chilli (init script included) to default runlevel (init script from above should have precedence); under SUSE, runlevels are manipulated with "insserv":
insserv chilli-init
insserv dhcpd
insserv mysql
insserv apache2
insserv freeradius
insserv chilli
4. Attached for convenience
init script for VLANs, IP
init script for chilli daemon
chilli.conf (comments stripped)
dhcpd.conf (comments stripped)
modified iptables ruleset
/etc/raddb files (comments stripped)
sample Lancom AP config (shared secrets, IP info stripped)