<u>Home</u> > <u>Network and technology service docs</u> > <u>eduroam</u> > <u>Technical Reference Docs</u> > Using certificates issued by the Janet Certificate Service with MS IAS

Using certificates issued by the Janet Certificate Service with MS IAS

Using certificates issued by the Janet Certificate Service with MS IAS

Jezz Palmer, University of Wales Swansea Paul Matthews, University of Wales Swansea Josh Howlett, Janet

Table of Contents

- 1. Introduction
- 2. Procedure
- 2.1 Patch OpenSSL
- 2.2 Generate the certificate
- 2.3 Configure the Microsoft Management Console
- 2.4 Import the chaining certificate
- 2.5 Import the IAS certificate
- 2.6 Configuring the IAS Remote Access Policies
- 3. Appendix I Example ias.cnf
- 4. Appendix II Acknowledgements
- 1. Introduction

This document explains how to configure Microsoft Internet Authentication Service to use certificates issued by the Janet Certificate Service (JCS).

The material in this document is based on Stephen Pillinger's documentation (<u>http://www.cs.bham.ac.uk/~smp/projects/peap/</u>[1]) and has been extended to explain how a Certificate Signing Request (CSR) can be signed by a third-party Certificate Authority (CA), such as Comodo CA, issued through the Janet Certificate Service.

Ensure that your organisation is registered with the Janet Certificate Service before proceeding with these instructions.

2. Procedure

2.1 Patch OpenSSL

- 1. Create a working directory and change directory into it.
 - \$ mkdir openssl-peap && cd openssl-peap
- 2. Download the openssl source tarball and untar it.
 - \$ wget http://openssl.org/source/openssl-0.9.8e.tar.gz
 - \$ tar -xzf openssl-0.9.8e.tar.gz
- 3. Download Stephen Pillinger's patch.

```
$ wget http://www.cs.bham.ac.uk/~smp/projects/peap/openssl-
0.9.8a-patch.txt
```

- 4. Change directory to the *openssl* source directory, patch it and compile.
 - \$ patch -p1 < ../openssl-0.9.8a-patch.txt</pre>
 - \$./config && make

Note: the patched version of openssl is built in the apps directory. This documentation assumes the use of this binary where "openssl" is mentioned. Ensure that this binary is used, not another openssl binary that may be present on the system (such as that provided by the operating system vendor).

2.2 Generate the certificate

5. Create the server key, server.key.pass.

```
$ openssl genrsa -des3 -out server.key.pass -passout pass:1234
2048
```

6. Remove the password from server.key.pass, creating server.key.

```
$ openssl rsa -in server.key.pass -out server.key -passin
pass:1234
```

7. Delete server.key.pass.

\$ rm server.key.pass

8. Generate the certificate request file, server.csr.

Note: the -config option points to ias.cnf, an openssl configuration that specifies the use of the required extendedKeyUsage attributes. An example is provided in Appendix I.

```
$ openssl req -new -days 3650 -config ias.cnf -key server.key -
out server.csr
```

Note: do not enter a challenge password if prompted.

9. Display server.csr.

```
$ openssl req -noout -text -in server.csr
Certificate Request:
   Data:
     Version: 0 (0x0)
     Subject: C=GB, ST=Wales, L=Swansea, O=University of Wales, Swansea,
OU=LIS.
CN=radius.swan.ac.uk/emailAddress=postmaster@swansea.ac.uk [2]
     Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
        RSA Public Key: (2048 bit)
          Modulus (2048 bit):
            00:d5:02:6d:a1:6c:15:fa:d5:97:4a:58:c0:33:25:
            95:3a:d1:ad:db:49:86:34:8b:d6:52:32:d5:12:d3:
            c4:04:a8:ba:b5:bc:7e:ed:12:ac:67:8e:0b:b0:57:
            67:12:3c:ba:98:b1:2c:b7:f2:9d:3a:60:ff:89:80:
            4a:06:63:a2:f7:50:8f:aa:e2:0f:74:1d:8c:2c:10:
            81:ae:58:14:61:85:d7:01:19:63:b0:5c:28:be:ab:
            63:cd:dd:de:31:e8:72:6a:d4:fb:48:b5:36:cf:e3:
            d4:99:91:4c:76:d1:48:2e:7c:0f:ea:65:40:7e:f1:
             51:e6:d0:8d:07:7e:47:68:8d
          Exponent: 65537 (0x10001)
     Attributes:
     Requested Extensions:
        X509v3 Subject Key Identifier:
         D2:BC:E1:12:aC:38:EC:aa:A5:E9:9B:01:12:23:8B:30:7B:7a:15:0a
        X509v3 Key Usage:
         Digital Signature, Key Encipherment
        X509v3 Extended Key Usage:
         TLS Web Client
Authentication, TLS Web Server Authentication
   Signature Algorithm: sha1WithRSAEncryption
     34:f5:96:50:c8:15:3d:3b:c1:c6:6e:ef:a5:c1:ab:96:8e:79:
```

35:5f:36:54:92:6d:86:31:85:b9:23:d2:92:15:a5:d1:b7:9d: 71:4f:c3:63:ac:af:01:a5:56:8e:ce:de:8f:fa:2a:b4:fb:06: 7f:5f:12:d7:57:25:e2:ed:a3:90:d6:0c:df:b1:ed:53:69:8c: 16:f2:ba:e9:68:c3:84:7e:a5:66:09:a0:76:f6:9a:8e:79:38: 46:7c:69:56:1e:62:8a:41:5f:ce:bd:34:a8:4f:cf:31:f4:09: 1e:a6:4b:ba:26:5a:4f:70:aa:2b:3a:d6:17:87:d6:f9:5d:cd: 3c:8e

10. Submit the *server.csr* file through the Janet Certificate Service, by following the URL to the service's web portal: <u>https://certificates.ja.net/jcs</u> [3]

Note: Comodo will only issue a successful certificate request once the email challenge process has been completed, known as Domain Control Validation (DCV). The email address for this step is also selected through the web portal. For further information about DCV see https://community.jisc.ac.uk/library/janet-services-documentation/domain-control-validation-dcv-process-0

11. Once your application has been processed successfully, Comodo CA will email you with the requested certificate. Save the PEM file signed and provided by Comodo CA to a file called *Comodo-CA.pem*.

12. Rename the original server.key to cert.pem.

\$ cp server.key cert.pem

13. Generate a public and private key pair using the original server key and the signed Comodo-CA PEM file, *cert.pem*.

\$ openssl x509 -in Comodo-CA.pem >> cert.pem

14. Using the cert.pem file, generate a PKCS12 file.

```
$ openssl pkcsl2 -name "IAS Cert" -export -in cert.pem -out
cert.pl2 -CSP 'Microsoft RSA SChannel Cryptographic Provider' -
LMK
```

15. Copy your newly created certificate file, *cert.p12*, to the IAS server.

16. Download the TERENA SSL CA chaining certificate to the IAS server; this certificate is located at http://www.terena.org/activities/tcs/repository/TERENA_SSL_CA.pem [5]

2.3 Configure the Microsoft Management Console

Note: it is only necessary to complete this section if you have not already configured the Microsoft Management Console (MMC) for certificate management.

17. To open the MMC, select *Start...Run* and enter *mmc* and then press *Return*.

18. Within the new console window, select *File...Add/Remove Snap-in*.

19. Select the Add button and then select Certificates from the list of Available Standalone Snap-ins

20. Select the Computer account option and click Next.

21. Ensure that the *Local computer* option is selected, and then select *Finish*.

22. Select Close in the Add Standalone Snap-in dialogue.

23. Select OK on the Add/Remove Snap-in dialogue.

24. Select File...Save As, provide a name and location and select Save.

2.4 Import the chaining certificate

25. Open the Certificate Management Console if it isn't already open.

26. Expand the Certificates (Local Computer) tree.

27. Right-click the *Intermediate Certification Authorities*, select *All Tasks* and then select *Import*; this opens the *Certificate Import Wizard*.

28. Select *Next* and then browse to the 'TERENASSLCA.crt' chaining certificate; select it and then select *Open*.

29. Select *Next*, ensuring that the *Place all Certificates in the following store* option is selected, and that *Intermediate Certification Authorities* is listed in the *Certificate Store* box.

30. Select Next to accept, then Finish and finally OK to finish the Certificate Import Wizard.

2.5 Import the IAS certificate

31. Open the Certificate Management Console if it isn't already open.

32. Expand the Certificates (Local Computer) tree.

33. Right-click on *Personal*, select *All Tasks*, and then select *Import*; this opens the *Certificate Import Wizard*.

34. Select *Next* and then browse to the location of the IAS certificate; change the *Files of type* drop-down menu to *Personal Information Exchange (*.pfx, *.p12)*; select your IAS certificate and then select *Open*.

35. Select Next; do not enter a password and select Next.

36. Select *Next*, ensuring that the *Place all Certificates in the following store* option is selected, and that *Personal* is listed in the *Certificate Store* box.

37. Select Next to accept, then Finish and finally OK to finish the Certificate Import Wizard.

38. Within the Certificates Management Console, expand the *Certificates (Local Computer)* tree.

39. Expand the *Personal* branch and select *Certificates*.

40. Double-click the IAS certificate to open the certificate's *Properties* dialogue.

41. Select the *Details* tab, and then select the *Edit Properties* button.

42. By default the *Enable all purposes for this certificate* is selected; select the *Enable only the following purposes* option and ensure all are ticked.

43. Click *OK* to close the *Certificate Properties* dialogue, and then click *OK* to close the *'Certificate'* dialogue.

2.6 Configuring the IAS Remote Access Policies

44. Open the Internet Authentication Service management console.

45. Select the Remote Access Policies branch.

46. Double-click the policy that you wish to use for EAP-PEAP authentication.

47. Select the *Edit Profile* button and chose the *Authentication* tab.

48. Ensure that the *MS-CHAPv2* authentication method is selected, and then select the *EAP-Methods* button.

49. If *Protected-EAP (PEAP)* isn't already listed in the *EAP Types*: select the *Add* button and then select *Protected-EAP (PEAP)* and then select *OK*.

50. If *Protected-EAP (PEAP)* is listed in the *EAP Types* box, then select it and then select the *Edit* button.

51. Select the IAS certificate from the *Certificate Issued* drop-down menu and then select *Ok* to close the dialogue.

Note: if you can't see your certificate here, or get an error stating that there are no certificates available, try restarting the IAS service to force a reload of the certificate store.

52. Select Ok to close the Select EAP Providers dialogue.

53. Select Ok to close the Edit Dial-in Profile dialogue.

54. Select Ok to close the Remote Access Policy properties dialogue.

Appendix I – Example ias.cnf

The following text is an example *openssl* configuration file.

```
server.csr
#
# OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#
# This definition stops the following lines choking if HOME isn't
# defined.
HOME = .
```

```
RANDFILE
```

= \$ENV::HOME/.rnd

[req] default_bits default_keyfile distinguished_name attributes x509_extensions req_extensions	= 2048 = privkey.pem = req_distinguished_name = req_attributes = extensions = req_extensions
string_mask	= nombstr
[req_distinguished_name] countryName countryName_default countryName_min countryName_max	= Country Name (2 letter code) = GB = 2 = 2
stateOrProvinceName	= State or Province Name (full name)
stateOrProvinceName_default	= Wales
localityName	= Locality Name (eg, city)
localityName_default	= Swansea
0.organizationName	= Organization Name (eg, company)
0.organizationName_default	= University of Wales, Swansea
organizationalUnitName	= Organizational Unit Name (eg, section)
organizationalUnitName_default	= LIS

4. Appendix II – Acknowledgements	
commonName	= Common Name (eg, YOUR name)
StepheonRiaimgementiversity of Birmingham.	= 64
commonName_default Jan Tomasek, CESNET.	= radius.swan.ac.uk
Martin Kafara, EMWAC Group. emailAddress	= Email Address
Dispiaydasesnglefoutumn?:	= postmaster@swansea.ac.uk [2]
emailAddress_max Source URL: https://community.jisc.ac.uk/library/janet	= 64 -services-documentation/using-certificates-issued-
janet-certificate-service-ms-ias	C C
Llines attributes]	= A challenge password
[thatten/wereassbhord.ac.uk/~smp/projects/peap/	= 4
[2handhogenstaggoon@swansea.ac.uk	= 20
13 https://epuilsaworla.net/cs 141 https://community.ijsc.ac.uk/library/janet-services-d	ocumentation/domain-control-validation-dcv-
process-0	
[5] http://www.terena.org/activities/tcs/repository/TERE	NA_SSL_CA.pem
unstructuredName	= An optional company name
[extensions]	= hash
subjectKeyIdentifier	= keyid,issuer:always
authorityKeyIdentifier	= digitalSignature,keyEncipherment
keyUsage	= clientAuth,serverAuth
extendedKeyUsage	

[req_extensions]	
subjectKeyIdentifier	
keyUsage	
extendedKeyUsage	

= hash

- = digitalSignature,keyEncipherment = clientAuth,serverAuth