cockpit
IT Service Manager

# SSL certificate installation

## Technical specification

# Table of contents

# Introduction

This document describes the stages to be followed in order to install SSL certificate on Linux or Windows operating system with Apache web server or Tomcat web server.
You will have to request a certificate to a certification authority (Thawte, Verisign...).

# Linux operating system

## I. Software

| Usage | Software | Version / Module |
|---|---|---|
| Operating system | Linux | Ubuntu server<br>Debian<br>CentOS |
| Web server | Apache Server | 2.X<br>- libapache-mod-ssl |
| Web server | Tomcat | Delivered with Cockpit IT Service Manager |

## II. Files generation

### A. Generate private key (RSA)

Create a specific directory for your Apache certificates.

For example: /etc/apache2/cert/

From your specific certificates directory, generate your private key.

For a 2048 bits RSA private key, use next command:

*sudo openssl genrsa -out mydomain.com.key -aes256 2048*

Example with cockpit-itsm.com domain:

*sudo openssl genrsa -out cockpit-itsm.com.key -aes256 2048*

Enter a pass phrase. Don't forget to save it, you will have to use it for starting Apache server.

```
Generating RSA private key, 2048 bit long modulus
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for cockpit-itsm.com.key:
Verifying - Enter pass phrase for cockpit-itsm.com.key:
```

A ".key" file is generated, it must be like next example:

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-256-CBC,4BCA4E1FB3BDBF693D80CEBFBF1A2937
```

```
GkD2mP29McWQ3wPqV0Q+vb+F48jaPWsR6APE7z5M6efRt0XQct62BrsMoRWN9V5f
wtmMhkDHUFdoZ2guEvZ4vo/+vIc/+gKsFPdtldZUiY+0T9RZr5LzvGyPuA6sr1Cq
...
bWAGuRo0M1n4DajCvZV46iEnr4Rv3B8Vc4z/xLRZfJ1zaOzNng+kshjX/FlkUhwH
7qNivkUjpD2t2piY4ul4UaN5+7q7KpRfFezCWJH477mLPstKETZfpEJCjhbQFuOa
-----END RSA PRIVATE KEY-----
```

## B. Generate certificate signature request (CSR)

Generate your certificate signature request, you will have to send it to your certification authority.

From your specific certificates directory, generate your CSR file. Use next command:

*sudo openssl req -nodes -newkey rsa:2048 -keyout mydomain.com.key -out mydomain.com.csr*

Example with cockpit-itsm.com domain:

*sudo openssl req -nodes -newkey rsa:2048 -keyout cockpit-itsm.com.key -out cockpit-itsm.com.csr*

Enter your pass phrase and fill in fields. Filled in information must be correct, it will be checked by your certification authority.

```
Enter pass phrase for cockpit-itsm.com.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:ILE DE FRANCE
Locality Name (eg, city) []:PARIS
Organization Name (eg, company) [Internet Widgits Pty Ltd]: COCKPIT ITSM
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []: *.cockpit-itsm.com
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

A ".csr" file is generated, it must be like next example:

```
-----BEGIN CERTIFICATE-----
MIIFLjCCBBagAwIBAgIQIKyBC22h97RRqBcnRy8RTzANBgkqhkiG9w0BAQUFADCB
...
QxNHrwr+ubvPb2+/lvgHLV/rwU0jZr2FLRnc51a/g7vk2aFdE0d+ipKJUzpv7St8
rBR6hvGUbQp201DukK9gnMrPrBoi9gbfJ5gSTLrpIlpnMQ==
```

```
-----END CERTIFICATE-----
```

Send CSR file to your certification authority.

Follow the certification authority process.

Put delivered certificate (x509 format) on your server (on your specific certificates directory).

Use next format name: mydomain.com.crt

From your specific certificates directory, check your certificate.

Use next command:

*sudo openssl x509 -text -noout -in mydomain.com.crt*

Example with cockpit-itsm.com domain:

*sudo openssl x509 -text -noout -in cockpit-itsm.com.crt*

### C. Get Certificate Authorities file

Apache need a Certificate Authorities file.

Use your Certificate Authority help for finding it.

For example, an Thawte web site, Certificate Authorities file is downloadable on:

https://www.thawte.com/roots/index.html

## III. Apache configuration

Add next parameters on your Apache configuration file.

```
SSLEngine on
SSLCertificateFile /yourcerticatespath/mydomain.com.crt
SSLCertificateKeyFile /yourcerticatespath/mydomain.com.key
SSLCACertificateFile /yourcerticatespath/ca_file.crt
```

You can use a virtual host for your SSL configuration. For example:

```
<VirtualHost XXX.XXX.XXX.XXX:443>
        ServerName mydomain.com
        <Proxy *>
                AddDefaultCharset Off
                Order deny,allow
                Allow from all
        </Proxy>
        SSLEngine On
        SSLProxyEngine On
        SSLCACertificateFile /yourcerticatespath/ca_file.crt
        SSLCertificateFile /yourcerticatespath/mydomain.com.crt
        SSLCertificateKeyFile yourcerticatespath/mydomain.com.key
</VirtualHost>
```

## IV. Tomcat configuration

## A. Keystore file generation

In the next step, we need to create a PKCS12 keystore for use by the portal. If your SSL certificate authority does not provide PKCS12 keystore files, you can generate it with the following procedure.

Prerequisites

We assume the following files are present in the current working directory:
server.crt - PEM format file of CAs
server.key - private key
server.ca-bundle - CA bundle file

The following command will ask you for a keystore password. Please use the same password each time you are asked for it. Don't forget to update the file server.xml with this password if you have not done so already.

Debian / Ubuntu:

*sudo cat server.key server.crt server.ca-bundle | openssl pkcs12 -export -nodes -name tomcat -out /home/koaly/exp/portal/conf/tomcat.p12*

RedHat / CentOS:

*cat server.key server.crt server.ca-bundle | openssl pkcs12 -export -nodes -name tomcat -out /home/koaly/exp/portal/conf/tomcat.p12*

## B. Cockpit IT Service Manager - Portal configuration

By default, the portal is not configured for SSL. To switch to SSL, you can use the following procedure.

Open the file /home/koaly/exp/portal/conf/server.xml:
Debian / Ubuntu:

*sudo vi /home/koaly/exp/portal/conf/server.xml*

RedHat / CentOS:

*vi /home/koaly/exp/portal/conf/server.xml*

Replace the following lines

```
<!-- HTTP (No SSL): Uncomment the following 4 lines -->
<Connector executor="tomcatThreadPool" URIEncoding="UTF-8" server="Koaly EXP Portal"
        port="8080" protocol="HTTP/1.1"
        connectionTimeout="20000"/>

<!-- HTTP (SSL): Uncomment the following 6 lines -->
<!--<Connector executor="tomcatThreadPool" URIEncoding="UTF-8" server="Koaly EXP Portal"-->
        <!--port="8443" protocol="HTTP/1.1" SSLEnabled="true"-->
        <!--scheme="https" secure="true"-->
        <!--clientAuth="false" sslProtocol="TLS"-->
```

```
        <!--keystoreFile="${catalina.base}/conf/tomcat.ks"-->
        <!--keystorePass="koaly2009"/>-->
```

with

```
    <!-- HTTP (No SSL): Uncomment the following 4 lines -->
    <!--
    <Connector executor="tomcatThreadPool" URIEncoding="UTF-8" server="Koaly EXP Portal"
            port="8080" protocol="HTTP/1.1"
            connectionTimeout="20000"/>
    -->

    <!-- HTTP (SSL): Uncomment the following 6 lines -->
    <Connector executor="tomcatThreadPool" URIEncoding="UTF-8" server="Koaly EXP Portal"
            port="8443" protocol="HTTP/1.1" SSLEnabled="true"
            scheme="https" secure="true"
            clientAuth="false" sslProtocol="TLS"
            keystoreType="PKCS12"
            keystoreFile="${catalina.base}/conf/tomcat.p12"
            keystorePass="{password}"/>
```

Note: The {password} is the password of your PKCS12 keystore

### C. Local port forward

Use iptables to forward traffic from port 443 to port 8443

Note: In the following commands, *eth0* must be replaced by the network interface you want to redirect traffic for.

Debian / Ubuntu:

We use iptables-persistent to make the current iptables persistent across reboots. If the package is not installed yet, you must install it first, accepting to persist the current rules:

*sudo apt-get -y install iptables-persistent*

Install the redirection and make the current state persistent:

*sudo iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 443 -j REDIRECT --to-port 8443*
*sudo invoke-rc.d iptables-persistent save*

RedHat / CentOS:

*vi /etc/sysconfig/iptables*

Add the following line to the *filter table:

```
-A INPUT -p tcp -m state --state NEW -m tcp --dport 8443 -j ACCEPT
```

Add the following parameters at the end of the file (or add the highlighted line if the nat table already exists):

```
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A PREROUTING -i eth0 -p tcp -m tcp --dport 443 -j REDIRECT --to-ports 8443
COMMIT
```

Restart the firewall:

*service iptables restart*

# Windows operating system

## I. Software

| Usage | Software | Version / Packages |
|---|---|---|
| Operating system | Windows | 2012 server |
| Web server | Apache Server | Apache 2.2.X<br>- libapache-mod-ssl |
| Web server | Tomcat | Delivered with Cockpit IT Service Manager |
| SSL | OpenSSL | |

## II. Files generation

### A. Generate private key (RSA)

You need to know where your openssl program is located.

From your openssl directory, generate your private key.

For a 2048 bits RSA private key, use next command:

*sudo openssl genrsa -out mydomain.com.key -aes256 2048*

Example with cockpit-itsm.com domain:

*sudo openssl genrsa -out cockpit-itsm.com.key -aes256 2048*

Enter a pass phrase. Don't forget to save it, you will have to use it for starting Apache server.

```
Generating RSA private key, 2048 bit long modulus
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for cockpit-itsm.com.key:
Verifying - Enter pass phrase for cockpit-itsm.com.key:
```

A ".key" file is generated, it must be like next example:

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-256-CBC,4BCA4E1FB3BDBF693D80CEBFBF1A2937

GkD2mP29McWQ3wPqV0Q+vb+F48jaPWsR6APE7z5M6efRt0XQct62BrsMoRWN9V5f
wtmMhkDHUFdoZ2guEvZ4vo/+vIc/+gKsFPdtldZUiY+0T9RZr5LzvGyPuA6sr1Cq
...
```

```
bWAGuRo0M1n4DajCvZV46iEnr4Rv3B8Vc4z/xLRZfJ1zaOzNng+kshjX/FlkUhwH
7qNivkUjpD2t2piY4ul4UaN5+7q7KpRfFezCWJH477mLPstKETZfpEJCjhbQFuOa
-----END RSA PRIVATE KEY-----
```

### B. Create an openssl configuration file

On your openssl directory, create a new openssl.conf file.

Copy next parameters on this file:

```
#
# SSLeay example configuration file.
# This is mostly being used for generation of certificate requests.
#

RANDFILE              = .rnd

####################################################################
[ ca ]
default_ca    = CA_default          # The default ca section

####################################################################
[ CA_default ]

dir           = demoCA              # Where everything is kept
certs         = $dir\certs          # Where the issued certs are kept
crl_dir       = $dir\crl            # Where the issued crl are kept
database      = $dir\index.txt      # database index file.
new_certs_dir   = $dir\newcerts     # default place for new certs.

certificate   = $dir\cacert.pem     # The CA certificate
serial        = $dir\serial         # The current serial number
crl           = $dir\crl.pem        # The current CRL
private_key   = $dir\private\cakey.pem   # The private key
RANDFILE      = $dir\private\private.rnd # private random number file

x509_extensions = x509v3_extensions     # The extentions to add to the cert
default_days    = 365                 # how long to certify for
default_crl_days= 30                  # how long before next CRL
default_md      = md5                 # which md to use.
preserve        = no                  # keep passed DN ordering

# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy        = policy_match

# For the CA policy
[ policy_match ]
countryName           = optional
stateOrProvinceName   = optional
organizationName      = optional
```

```
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

####################################################################
[ req ]
default_bits            = 1024
default_keyfile         = privkey.pem
distinguished_name      = req_distinguished_name
attributes              = req_attributes

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)

localityName                    = Locality Name (eg, city)

0.organizationName              = Organization Name (eg, company)

organizationalUnitName          = Organizational Unit Name (eg, section)

commonName                      = Common Name (eg, your website's domain name)
commonName_max                  = 64

emailAddress                    = Email Address
emailAddress_max                = 40

[ req_attributes ]
challengePassword               = A challenge password
challengePassword_min           = 4
challengePassword_max           = 20

[ x509v3_extensions ]

# under ASN.1, the 0 bit would be encoded as 80
nsCertType                      = 0x40
```

```
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName
#nsCertSequence
#nsCertExt
#nsDataType
```

### C. Generate certificate signature request (CSR)

Generate your certificate signature request, you will have to send it to your certification authority.

From your openssl directory, generate your CSR file. Use next command:

*sudo openssl req -config openssl.conf -nodes -newkey rsa:2048 -keyout mydomain.com.key -out mydomain.-com.csr*

Example with cockpit-itsm.com domain:

*sudo openssl req -config openssl.conf -nodes -newkey rsa:2048 -keyout cockpit-itsm.com.key -out cockpit-itsm.com.csr*

Enter your pass phrase and fill in fields. Filled in information must be correct, it will be checked by your certification authority.

```
Enter pass phrase for cockpit-itsm.com.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:ILE DE FRANCE
Locality Name (eg, city) []:PARIS
Organization Name (eg, company) [Internet Widgits Pty Ltd]: COCKPIT ITSM
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []: cockpit-itsm.com
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

A ".crt" file is generated, it must be like next example:

```
-----BEGIN CERTIFICATE-----
MIIFLjCCBBagAwIBAgIQIKyBC22h97RRqBcnRy8RTzANBgkqhkiG9w0BAQUFADCB
izELMAkGA1UEBhMCVVMxFTATBgNVBAoTDHRoYXd0ZSwgSW5jLjE5MDcGA1UECxMw
...
QxNHrwr+ubvPb2+/lvgHLV/rwU0jZr2FLRnc51a/g7vk2aFdE0d+ipKJUzpv7St8
rBR6hvGUbQp201DukK9gnMrPrBoi9gbfJ5gSTLrpIlpnMQ==
-----END CERTIFICATE-----
```

Send CSR file to your certification authority.

Follow the certification authority process.

Put delivered certificate (x509 format) on your server (on your specific certificates directory).

Use next format name: mydomain.com.crt

From your openssl directory, check your certificate.

Use next command:

*sudo openssl x509 -text -noout -in mydomain.com.crt*

Example with cockpit-itsm.com domain:

*sudo openssl x509 -text -noout -in cockpit-itsm.com.crt*

### D. Get Certificate Authorities file

Apache need a Certificate Authorities file

Use your Certificate Authority help for finding it.

For example, an Thawte web site, Certificate Authorities file is downloadable on:

https://www.thawte.com/roots/index.html

### E. Create specific directory for certificates

Create a specific directory for your Apache certificates.

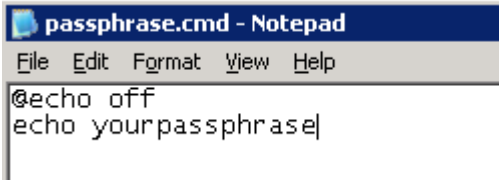
For example: c:\Program Files\Apache\cert\

Copy all certificates and keys files to this directory (mydomain.com.crt, mydomain.com.key, ca_file.crt)

### F. Create a script for Passphrase

On your specific directory for your Apache certificates, create a script "passphrase.cmd" which gives SSL passphrase for Apache.

*1.@echo off*
*2.echo yourpassphrase*



# III. Apache configuration

Add next parameters on your Apache configuration file.

```
SSLHonorCipherOrder On
SSLProtocol -ALL +TLSv1
SSLCipherSuite ECDHE-RSA-AES256-SHA384:AES256-SHA256:RC4:HIGH:!MD5:!aNULL:!EDH:!
AESGCM;
SSLHonorCipherOrder on

SSLEngine on
SSLCertificateFile /yourcerticatespath/mydomain.com.crt
SSLCertificateKeyFile /yourcerticatespath/mydomain.com.key
SSLCACertificateFile /yourcerticatespath/ca_file.crt
SSLPassPhraseDialog exec:/yourcerticatespath/passphrase.bat
Header Add Strict-Transport-Security "max-age=15768000"
```

You can use a virtual host for your SSL configuration. For example:

```
SSLHonorCipherOrder On
SSLProtocol -ALL +TLSv1
SSLCipherSuite ECDHE-RSA-AES256-SHA384:AES256-SHA256:RC4:HIGH:!MD5:!aNULL:!EDH:!
AESGCM;
SSLHonorCipherOrder on

<VirtualHost XXX.XXX.XXX.XXX:443>
        ServerName mydomain.com
        <Proxy *>
                AddDefaultCharset Off
                Order deny,allow
                Allow from all
        </Proxy>
        SSLEngine On
        SSLProxyEngine On
        SSLCACertificateFile /yourcerticatespath/ca_file.crt
        SSLCertificateFile /yourcerticatespath/mydomain.com.crt
        SSLCertificateKeyFile yourcerticatespath/mydomain.com.key
        SSLPassPhraseDialog exec:/yourcerticatespath/passphrase.bat
        Header Add Strict-Transport-Security "max-age=15768000"
</VirtualHost>
```

# IV. Tomcat configuration

## A. Keystore file generation

In the next step, we need to create a PKCS12 keystore for use by the portal. If your SSL certificate authority does not provide PKCS12 keystore files, you can generate it with the following procedure.

**Prerequisites**

You will need access to an installation of OpenSSL. If you do not want to install OpenSSL on the server, you can install it on any other machine and copy the generated file to the target directory. If you have access to a Linux machine, you can use this as well. Openssl is pre-installed on all major Linux distributions.

We assume the following files are present in the current working directory:
server.crt - PEM format file of CA
server.key - private key
server.ca-bundle - CA bundle file

Concatenate the files to provide a full certification path:
*copy server.key+server.crt+server.ca-bundle target.key*

The following command will ask you for a keystore password. Please use the same password each time you are asked for it. Don't forget to update the file server.xml with this password if you have not done so already.

Open a command prompt and execute the following command (openssl.exe must be on your PATH):
*openssl pkcs12 -export -in target.key -nodes -name tomcat -out tomcat.p12*

To finish, move the file tomcat.p12 to the directory c:\koaly\exp\portal\conf\ on your server.

## B. Cockpit IT Service Manager - Portal configuration

By default, the portal is not configured for SSL. To switch to SSL, you can use the following procedure.
*Open the file "C:\koaly\exp\portal\conf\server.xml" in a text editor.*
Replace the following lines

```
        <!-- HTTP (No SSL): Uncomment the following 4 lines -->
    <Connector executor="tomcatThreadPool" URIEncoding="UTF-8" server="Koaly EXP Portal"
            port="80" protocol="HTTP/1.1"
            connectionTimeout="20000"
            redirectPort="443"/>

    <!-- HTTP (SSL): Uncomment the following 7 lines -->
    <!--
    <Connector executor="tomcatThreadPool" URIEncoding="UTF-8" server="Koaly EXP Portal"
            port="443" protocol="HTTP/1.1" SSLEnabled="true"
            scheme="https" secure="true"
            clientAuth="false" sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"
ciphers="TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,TLS_ECDHE_RSA_WITH_RC4_128_SHA,TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA,TLS_RSA_WI
```

```
TH_AES_256_CBC_SHA256,TLS_RSA_WITH_AES_256_CBC_SHA,TLS_RSA_WITH_AES_128_CBC_SH
A256,TLS_RSA_WITH_AES_128_CBC_SHA,SSL_RSA_WITH_RC4_128_SHA"
                keystoreType="PKCS12"
                keystoreFile="${catalina.base}/conf/tomcat.p12"
                keystorePass="koaly2009"/>
        -->
```

with

```
        <!-- HTTP (No SSL): Uncomment the following 4 lines -->
        <!--
        <Connector executor="tomcatThreadPool" URIEncoding="UTF-8" server="Koaly EXP Portal"
                port="80" protocol="HTTP/1.1"
                connectionTimeout="20000"
                redirectPort="443"/>
        -->

        <!-- HTTP (SSL): Uncomment the following 6 lines -->
        <Connector executor="tomcatThreadPool" URIEncoding="UTF-8" server="Koaly EXP Portal"
                port="443" protocol="HTTP/1.1" SSLEnabled="true"
                scheme="https" secure="true"
                clientAuth="false" sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"
ciphers="TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,TLS_ECDHE_RSA_WITH_AES_256_CBC_S
HA,TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,T
LS_ECDHE_RSA_WITH_RC4_128_SHA,TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA,TLS_RSA_WI
TH_AES_256_CBC_SHA256,TLS_RSA_WITH_AES_256_CBC_SHA,TLS_RSA_WITH_AES_128_CBC_SH
A256,TLS_RSA_WITH_AES_128_CBC_SHA,SSL_RSA_WITH_RC4_128_SHA"
                keystoreType="PKCS12"
                keystoreFile="${catalina.base}/conf/tomcat.p12"
                keystorePass="{password}"/>
```

> Note: The {password} is the password of your PKCS12 keystore.

## V. Firewall

To allow the HTTPS traffic through the Windows firewall, you need to create a new rule. Open a command prompt as administrator and execute the following command:

*netsh advfirewall firewall add rule name="Allow HTTPS" dir=in action=allow protocol=TCP localport=443*

> Important: If UAC is active on your machine, just being logged on as administrator does not provide you the necessary rights. You need to right-click the Command Prompt item in the start menu and choose “Run as administrator”.

## Tests

Use next URL for testing your SSL certificate.

https://www.ssllabs.com/ssltest/index.html

# Document end