HP-UX - Stunnel Configuration

Article Number: 384 | Rating: Unrated | Last Updated: Fri, Sep 28, 2018 8:10 PM

HP-UX - Stunnel Configuration

Information

Stunnel configuration procedure.

Details

The steps outlined below will generate a new private key and certificate request for a host, send the request to the local certificate authority on ${cahost}, then sign and return the request from ${cahost}. Once that is done, the signed certificate is used in the stunnel configuration file to create an encrypted telnet tunnel.

Once the encrypted tunnel is tested/verified, the user will know that stunnel and the certificate are working correctly. After that, it is just a matter of getting the configuration for the Oracle database. This procedure will eliminate two very large variables in the troubleshooting process (stunnel and the certificate).

The additions also include information on the log entries to look for, to verify connectivity.

Compile/install stunnel or ID where it is currently installed. On HP systems, check under **/opt/hpws/apache/stunnel/sbin**.

Create a directory for the certificates. CD into it.

mkdir -p -m 700 /root/certs; cd $_

Create a private key and cert request. Supply an aribitrary passphrase; we'll remove it later:
openssl req -newkey rsa:1024 -keyout ${host}_private.pem -keyform PEM -out ${host}_req.pem
Generating a 1024 bit RSA private key
....++++++
.....++++++
writing new private key to '${cahost}_private.pem'
Enter PEM pass phrase: <==== Enter passphrase here
Verifying - Enter PEM pass phrase: <=== Reenter same passphrase here
-----
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]:US
State or Province Name (full name) [Some-State]:IL
Locality Name (eg, city) []:Naperville
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MYCO
Organizational Unit Name (eg, section) []:IT
Common Name (eg, YOUR name) []:FQDN of the host
Email Address []:YOUR email address

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:Any arbitrary word; it's not used
An optional company name []:MYCO

Send the request to the CA directory on ${cahost}
scp ${host}_req.pem ${cahost}:${caroot}

Sign the certificate.

Log into ${cahost} as root, cd ${caroot}.

The CA needs specific items in the **openssl.conf** file. Since these entries are not required for normal openssl operation, the user needs to use a different conf file.

export OPENSSL_CONF=${caroot}/openssl.conf

Sign the request:

openssl ca -in ${host}_req.pem
Using configuration from ${caroot}/openssl.conf
Enter pass phrase for ${caroot}/private/cakey.pem: ${passphrase for ${cahost}'s private key}
DEBUG[load_index]: unique_subject = "yes"
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName :PRINTABLE:'US'
stateOrProvinceName :PRINTABLE:'IL'
localityName :PRINTABLE:'Naperville'
organizationName :PRINTABLE:'MYCO'
organizationalUnitName:PRINTABLE:'IT'
commonName :PRINTABLE:'${cahost}.myco.com'
emailAddress :IA5STRING:'oleary@myco'
Certificate is to be certified until Jul 11 13:54:58 2009 GMT (365 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Certificate:
Data:
Version: 3 (0x2)
Serial Number: 3 (0x3)
Signature Algorithm: md5WithRSAEncryption
Issuer: CN=MYCO, ST=IL, C=US/emailAddress=oleary@myco, O=Root Certificate Authority
Validity
Not Before: Jul 11 13:54:58 2008 GMT
Not After : Jul 11 13:54:58 2009 GMT
Subject: CN=${cahost}.myco.com, ST=IL, C=US/emailAddress=oleary@myco, O=MYCO, OU=IT
Subject Public Key Info:

Public Key Algorithm: rsaEncryption
RSA Public Key: (1024 bit)
Modulus (1024 bit):
00:c8:f2:4d:5b:a4:8d:c9:37:66:4f:77:3d:a4:b4:
2e:68:84:21:e5:64:37:56:ea:e5:f0:9c:72:f5:ba:
1c:65:21:9a:84:21:36:06:24:86:92:56:e6:f5:14:
30:d6:d9:6f:ad:8d:e9:11:b8:49:e1:3e:d7:f8:3d:
57:9b:64:29:87:9b:9c:c3:ea:00:80:b5:03:be:72:
02:dc:75:56:81:59:04:bb:e7:8e:53:56:16:0c:09:
97:a7:ea:0a:c7:e6:55:14:cb:92:1b:79:d6:1f:dc:
96:49:b7:ae:7a:d1:67:73:ff:6a:ef:69:e0:15:ef:
1b:67:b1:90:c2:2c:fe:96:c5
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Basic Constraints:
CA:FALSE
Signature Algorithm: md5WithRSAEncryption
61:c7:0b:2e:e8:65:1b:d5:7b:b0:1a:60:4b:f0:07:b7:d6:b6:
9d:21:ea:b7:d5:3a:41:85:56:7f:ca:ed:ad:9d:06:d7:52:8c:
9e:13:2a:ef:0f:1f:32:a7:72:12:5d:e0:00:0e:ee:44:2f:ad:
83:06:c0:ce:94:b3:5e:5e:34:c9:eb:d1:bc:dc:bf:71:6c:e0:
b5:65:0c:1d:64:60:e7:98:31:d4:93:18:7d:d7:4f:fc:e2:e6:
8e:e0:06:a2:1d:ff:db:41:d0:c4:ff:8e:18:7c:8e:b8:95:1d:
d3:59:38:73:d1:92:c1:16:f5:91:8b:1e:a3:d6:cf:04:db:24:
84:34:28:c7:03:59:02:9c:ab:c4:31:06:dd:4b:8f:f0:64:09:
e3:7a:13:e0:57:cf:a9:d8:81:7d:05:87:d9:a3:c4:78:03:36:
fb:1e:58:65:a0:fd:ff:f4:8f:32:60:0a:dc:53:c8:4f:00:22:
43:d1:dd:7a:e6:a6:63:67:87:53:9a:7a:c5:be:f0:0c:92:74:
f5:d2:05:c6:51:60:3b:b3:83:53:40:7e:dd:44:a5:c6:32:63:
c6:99:c4:ea:c9:36:be:f5:e9:d3:98:27:eb:59:4b:52:4c:6f:
d2:d6:4d:df:2e:22:7c:7e:18:7b:88:8a:5a:41:6a:53:e6:6b:
60:48:38:b8
-----BEGIN CERTIFICATE-----
MIIC/TCCAeWgAwIBAgIBAzANBgkqhkiG9w0BAQQFADB3MQwwCgYDVQQDEwNBTFUx
CzAJBgNVBAgTAklMMQswCQYDVQQGEwJVUzEoMCYGCSqGSIb3DQEJARYZb2xlYXJ5
QGFsY2F0ZWwtbHVjZW50LmNvbTEjMCEGA1UEChMaUm9vdCBDZXJ0aWZpY2F0ZSBB
dXRob3JpdHkwHhcNMDgwNzExMTM1NDU4WhcNMDkwNzExMTM1NDU4WjCBgTEgMB4G
A1UEAxMXdXNpbGlnbjMubmRjLmx1Y2VudC5jb20xCzAJBgNVBAgTAklMMQswCQYD

VQQGEwJVUzEoMCYGCSqGSIb3DQEJARYZb2xlYXJ5QGFsY2F0ZWwtbHVjZW50LmNv
bTEMMAoGA1UEChMDQUxVMQswCQYDVQQLEwJJVDCBnzANBgkqhkiG9w0BAQEFAAOB
jQAwgYkCgYEAyPJNW6SNyTdmT3c9pLQuaIQh5WQ3Vurl8Jxy9bocZSGahCE2BiSG
klbm9RQw1tlvrY3pEbhJ4T7X+D1Xm2Qph5ucw+oAgLUDvnIC3HVWgVkEu+eOU1YW
DAmXp+oKx+ZVFMuSG3nWH9yWSbeuetFnc/9q72ngFe8bZ7GQwiz+lsUCAwEAAaMN
MAswCQYDVR0TBAIwADANBgkqhkiG9w0BAQQFAAOCAQEAYccLLuhlG9V7sBpgS/AH
t9a2nSHqt9U6QYVWf8rtrZ0G11KMnhMq7w8fMqdyEl3gAA7uRC+tgwbAzpSzXl40
yevRvNy/cWzgtWUMHWRg55gx1JMYfddP/OLmjuAGoh3/20HQxP+OGHyOuJUd01k4
c9GSwRb1kYseo9bPBNskhDQoxwNZApyrxDEG3UuP8GQJ43oT4FfPqdiBfQWH2aPE
eAM2+x5YZaD9//SPMmAK3FPITwAiQ9HdeuamY2eHU5p6xb7wDJJ09dIFxlFgO7OD
U0B+3USlxjJjxpnE6sk2vvXp05gn61lLUkxv0tZN3y4ifH4Ye4iKWkFqU+ZrYEg4
uA==
-----END CERTIFICATE-----
Data Base Updated

The generic CA puts the newly signed certificates under the ${caroot}/certs directory with a numeric filename. The new key should be the last one modified. ID and copy the newly signed cert back to the target host:

ls -lart ${caroot}/certs | tail -1

scp ${caroot}/certs/##.pem ${host}:/root/certs/${host}_signed.pem

Remove the passphrase from the private key:

Back on the target system, (not ${cahost}) run openssl command to remove the passphrase; update directories and filenames as appropriate:

openssl rsa -in ${host}_private.pem -out ${host}_np_private.pem
Enter pass phrase for ${host}_private.pem: <== Enter passphrase used when creating the key
writing RSA key

mv ${cahost}_np_private.pem ${cahost}_private.pem

Execute stunnel -h to identify where the binary thinks the **stunnel.conf** file should be. While not mandatory, it may make things easier in the long run to use that default config file.

stunnel-h

Syntax:
stunnel [filename] | -help | -version | -sockets
filename - use specified config file instead of /opt/hpws/apache/stunnel/etc/stunnel/stunnel.conf
-help - get config file help
-version - display version and defaults
-sockets - display default socket options

Create a new stunnel.conf. Update directories and filenames as appropriate:
cert = /root/certs/${host}_signed.pem
key = /root/certs/${host}_private.pem
client = no
pid = /var/run/stunnel.pid
# setuid = nobody
# setgid = other

output = /var/run/stunnel.log
# foreground = yes

[telnet]
accept = 9999
connect = 127.0.0.1:23

Run stunnel. If the user chooses a different location for the **stunnel.conf**, execute stunnel with the new config file:

stunnel ${config_file}
On another system, that you have access to:
create a new stunnel.conf in a non-standard location with the following information. Update host names and directories as appropriate:
client = yes
pid = /var/run/stunnel.pid

[telnet]
accept = 127.0.0.1:9999
connect = ${host}.myco.com:9999

stunnel ${conf_file}
telnet localhost 9999

The user should be presented with a login prompt to the target host - ${cahost} in this example.

If the user has successfully made it to this point, stunnel is compiled, installed, and configured correctly. The user will need to make additional configuration updates for the Oracle/jdbc connection; however, stunnel and the cert are no longer variables in any troubleshooting exercises.

When all the parties are ready, edit the stunnel config file, comment out the telnet stanza, and update the information as follows. Here is the ascii version of the table below. One note: /var/run, on HPs, tends to be 555 permissions. In order to get the pid file owned by the Oracle ID, **I mkdir -p -m 1777 /var/run/stunnel**, then update the pid as described below:

Comment Data
Same as test cert = /root/certs/${host}_signed.pem
Same as test key = /root/certs/${host}_private.pem
Same as test client = no
Updated location so oracle user can write to it. pid = /var/run/stunnel/stunnel.pid
Set to the oracle ID of the database in question. setuid = oracle
Set to the group ID of the database in question. setgid = dba
Generic comment # Some debugging stuff useful for troubleshooting
New debug = 7
Choose an appropriate log location; log will be owned by root output = /var/adm/syslog/stunnel.log
New socket = l:TCP_NODELAY=1
New socket = r:TCP_NODELAY=1
Generic comment # Authentication stuff
New verify = 1
Commented out from test # foreground = yes
Comment out the telnet stanza # [telnet]
Comment out telnet stanza # accept = 9999
Comment out telnet stanza # connect = 127.0.0.1:23

New stanza [jdbc]
New stanza: use appropriate port - usually database port + 5000 accept = 51524
New stanza: use appropriate port. connect = 127.0.0.1:1524

Log entries:
2008.07.22 09:24:16 LOG7[7072:1]: jdbc accepted FD=1 from 192.168.12.10:3144 <=== Where connection's from
2008.07.22 09:24:16 LOG7[7072:1]: FD 1 in non-blocking mode
2008.07.22 09:24:16 LOG7[7072:2]: jdbc started <===== Stanza
2008.07.22 09:24:16 LOG7[7072:2]: TCP_NODELAY option set on local socket
2008.07.22 09:24:16 LOG5[7072:2]: jdbc connected from 192.168.12.10:3144
Handshake and connection setup
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): before/accept initialization
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 read client hello A
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 write server hello A
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 write certificate A
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 write certificate request A
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 flush data
2008.07.22 09:24:16 LOG7[7072:2]: SSL alert (read): warning: no certificate
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 read client key exchange A
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 read finished A
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 write change cipher spec A
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 write finished A
2008.07.22 09:24:16 LOG7[7072:2]: SSL state (accept): SSLv3 flush data
2008.07.22 09:24:16 LOG7[7072:2]: 1 items in the session cache
2008.07.22 09:24:16 LOG7[7072:2]: 0 client connects (SSL_connect())
2008.07.22 09:24:16 LOG7[7072:2]: 0 client connects that finished
2008.07.22 09:24:16 LOG7[7072:2]: 0 client renegotiatations requested
2008.07.22 09:24:16 LOG7[7072:2]: 1 server connects (SSL_accept())
2008.07.22 09:24:16 LOG7[7072:2]: 1 server connects that finished
2008.07.22 09:24:16 LOG7[7072:2]: 0 server renegotiatiations requested
2008.07.22 09:24:16 LOG7[7072:2]: 0 session cache hits
2008.07.22 09:24:16 LOG7[7072:2]: 0 session cache misses
2008.07.22 09:24:16 LOG7[7072:2]: 0 session cache timeouts
2008.07.22 09:24:16 LOG6[7072:2]: Negotiated ciphers: AES256-SHA SSLv3 Kx=RSA Au=RSA Enc=AES(256) Mac=SHA1
Data transfer
2008.07.22 09:24:16 LOG7[7072:2]: FD 2 in non-blocking mode
2008.07.22 09:24:16 LOG7[7072:2]: jdbc connecting 127.0.0.1:1524
2008.07.22 09:24:16 LOG7[7072:2]: Remote FD=2 initialized
2008.07.22 09:24:16 LOG7[7072:2]: TCP_NODELAY option set on remote socket

2008.07.22 21:26:53 LOG7[7072:2]: SSL socket closed on SSL_read
2008.07.22 21:26:53 LOG5[7072:2]: Connection closed: 4224 bytes sent to SSL, 4242 bytes sent to socket
2008.07.22 21:26:53 LOG7[7072:2]: jdbc finished (0 left)
Connection closed