Using Stunnel with TestCaddy

Contents

Introduction

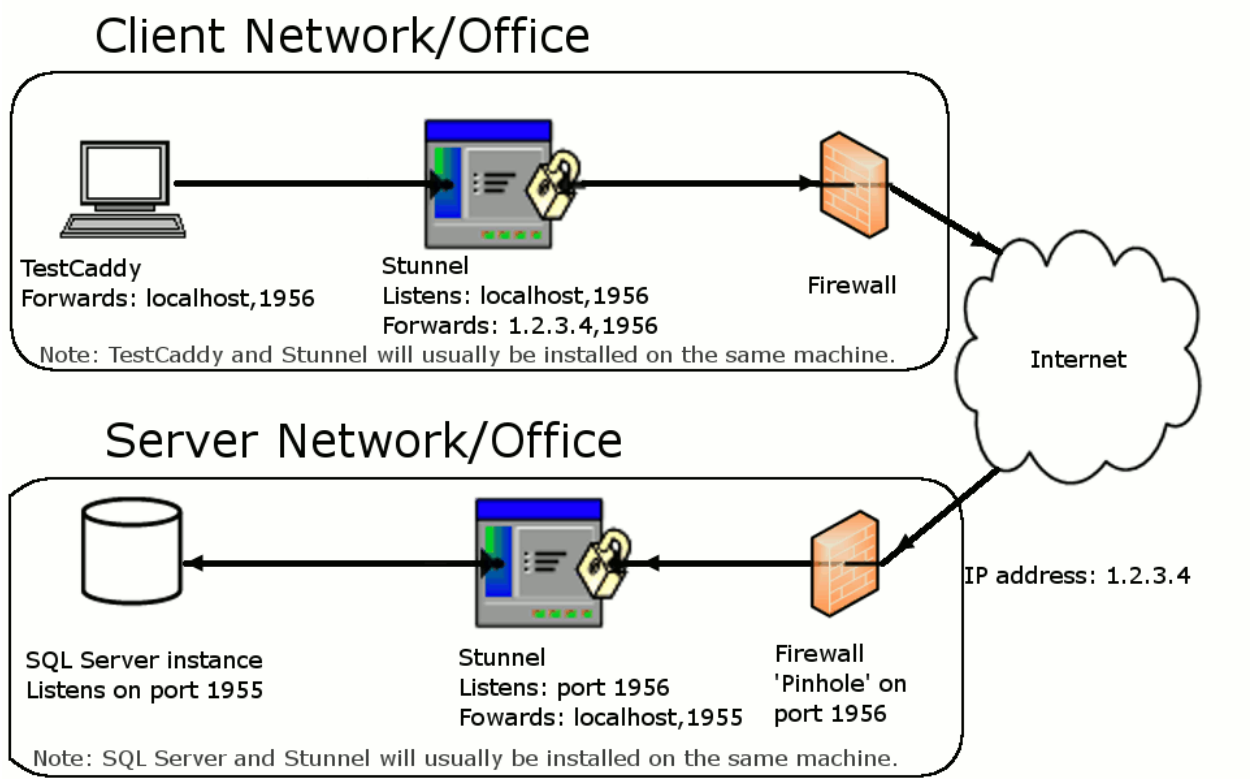
The following is an extract from the Stunnel official website: (www.stunnel.org)

Stunnel is a program that allows you to encrypt arbitrary TCP connections inside SSL (Secure Sockets Layer) available on both Unix and Windows. Stunnel can allow you to secure non-SSL aware daemons and protocols (like POP, IMAP, LDAP, etc) by having Stunnel provide the encryption, requiring no changes to the daemon's code.

Using Stunnel in conjunction with TestCaddy enables a secure point-to-point connection to a remote SQL Server over the internet. To do this, the following are required:

1. An instance of Stunnel must be installed on the server machine (where SQL Server is located)
2. An instance of Stunnel must be installed on the client machine (where TestCaddy is located or going to be installed once Stunnel is installed)
3. An SSL certificate for Stunnel to use, to ensure a secure connection
4. SQL Server must have a static port set.

To download Stunnel, please go to http://www.stunnel.org/download/



A simplified diagram of using Stunnel with TestCaddy

Configuring Stunnel to use with TestCaddy

Server configuration

- Download and install Stunnel on the machine where SQL Server is located.
- Before running Stunnel, it must be configured to enable a secure connection. This can be done by editing the 'stunnel.conf' file located in the Stunnel installation folder.

To modify, open stunnel.conf using Notepad, or your favourite text editor. If the lines mentioned below do not exist, add them, or otherwise change the existing lines.

Note: If any of the lines mentioned below begin with the ';' character in the stunnel.conf file then they are comments which will not take effect, so the ';' character at the beginning of the line must be removed.

1. Specify the certificate and files for Stunnel to use.

```
cert = host.pem
key = host.key
```

Where *host.pem* and *host.key* are certificate files (see below for details)

2. Specify authentication level. Setting it to 3 enforces the highest level of security.

```
; Authentication stuff
verify = 3
```

3. Specify certificate file

```
; It's often easier to use CAfile
CAfile = host.cert
```

4. Add the following. This is telling Stunnel to listen on the local port 1956, and forward to the local port 1955.

```
[sql]
;listen on port 1956
accept = 1956
; forward to port 1955 (this is the static port of the SQL Server)
connect = localhost:1955
```

5. Save and Exit.

Note: If you are only using Stunnel with TestCaddy, please make sure all the other lines begin with ';' with the exception of the following two lines:

```
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1
```

Client configuration

- Download and install Stunnel on the machine where TestCaddy is located.
- Before running Stunnel, it must be configured to enable a secure connection. This can be done by editing the 'stunnel.conf' file located in the Stunnel installation folder.

To modify, open stunnel.conf using Notepad, or your favourite text editor. If the lines mentioned below do not exist, add them, or otherwise change the existing lines.

Note: If any of the lines mentioned below begin with the ';' character in the stunnel.conf file then they are comments which will not take effect, so the ';' character at the beginning of the line must be removed.

1. Specify the certificate and files for Stunnel to use.

```
cert = host.pem
key = host.key
```

Where *host.pem* and *host.key* are certificate files (see below for details)

2. Specify authentication level. Setting it to 3 enforces the highest level of security.

```
; Authentication stuff
verify = 3
```

3. Specify certificate file

```
; It's often easier to use CAfile
CAfile = host.cert
```

4. Tell Stunnel that you are the client

```
; Use it for client mode
client = yes
```

5. Add the following. This is telling Stunnel to listen on the local port 1956, and forward to the specified IP address at the specified port. NOTE: You should specify the external IP address of the network where your SQL server is located.

```
[sql]
;listen on port 1956
accept = 1956
; forward to ip:port (this is the address of the Stunnel at the server machine)
connect = 1.2.3.4:1956
```

6. Save and Exit.

Note: If you are only using Stunnel with TestCaddy, please make sure all the other lines begin with ';' with the exception of the following two lines:

```
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1
```

SSL Certificate

A SSL certificate must be used with Stunnel to ensure a secure connection between TestCaddy and SQL Server. In addition to identification, your certificate is also used for encrypting the information travelling over the internet.

You can use your own certificate if you have one; alternatively you can create a self-signed certificate.

Creating self-signed certificate

To start with, you'll need OpenSSL. Download the recommended version from the OpenSSL site (http://www.openssl.org/), and extract it to a desired location (here we use C:\OpenSSL as an example). Note: We are not recommending an OpenSSL version as they change frequently. The process should be the same regardless of the version you use.

1. Open command prompt.
   Start > Programs > Accessories > Command Prompt

2. Navigate to the OpenSSL folder. For our example, we would type
   cd c:\openssl
   The dos prompt (beginning of the line) should now read C:\ OpenSSL>

3. Next, we're going to generate the key.
   openssl genrsa 1024 > host.key

4. Next, we're going to generate the certificate.
   openssl req -new -x509 -nodes -sha1 -days 365 -key host.key > host.cert
   Note: Here, we are making the certificate expire in 365 days. The shorter you make the expiry time, the safer it is, though this process must be repeated every time a certificate expires.

5. At this stage, you will be asked to fill out the certificate's X.509 attributes. The answers should be adjusted for the location. Note: When the certificate is going to be used in a peer to peer situation (as in this case), it does not really matter what information is in the certificate. Some minimal information should be supplied as in the following example.
   Country Name (2 letter code) [AU]:**NZ**
   State or Province Name (full name) [Some-State]:**Auckland**
   Locality Name (eg, city) []:**Auckland**
   Organization Name (eg, company) [Internet Widgits Pty Ltd]:**TamihaiLtd**
   Organizational Unit Name (eg, section) []:
   Common Name (eg, YOUR name) []:**mail.yourdomain.org**
   Email Address []:**postmaster@yourdomain.org**

6. Next, we're going to combine Key and Certificate files into *host.pem*.
   copy host.cert+host.key host.pem

You should now have three files (host.cert, host.key, host.pem) in C:\OpenSSL. These files need to be placed in the Stunnel folder on both the client and server machine, as earlier they were specified in the client and server config files.

Important notes about key and pem files

The data in the *host.key* and *key.pem* file must be protected, as anyone possessing the private key can read any data encrypted with this key! **Do not give these files to anyone, except to clients that need to connect to the remote SQL Server.**

Running Stunnel

Once Stunnel has been configured, the Stunnel service needs to be installed, and then Stunnel run. This needs to be done on both the Client and Server machines.

1. To install Stunnel service:
   ```
   Start > Programs > Stunnel > Service install
   ```

2. To run Stunnel service:
   ```
   Start > Programs > Stunnel > Service Start
   ```

Configuring Firewall for TestCaddy/Stunnel Access

A firewall can be used to restrict access to your network by forwarding only requests targeted at specific TCP/IP addresses in the local network. Requests for all other network addresses are blocked by the firewall.

Your IT administrator must configure the firewall which protects the network/office. A pinhole (port forwarding) must be added to the firewall, to forward incoming requests to the server box. For our example, open incoming port 1956, and forward to the server box on port 1956.

If you are running Windows Firewall on the SQL Server box itself, then you must also pinhole through the Windows Firewall.

Configuring Windows Firewall

To open a port in the Windows Firewall for TestCaddy to access the remote SQL Server

1. In Control Panel, open **Windows Firewall.**
2. In **Windows Firewall** dialog box, click the **Exceptions** tab, and then click **Add Port**.
3. In the **Add a Port** dialog box, in the **Name** text box, type **Stunnel/SQL**
4. In the **Port number** text box, type the port number that Stunnel is listening on
   E.g. 1965 for our example
5. Verify that **TCP** is selected, and then click **OK**.

Setting a static TCP/IP port for SQL Server

You need to configure SQL server to listen on a static TCP/IP port, i.e. one which does not change when the server restarts.

Note: Here, we mention SQL Server 2005; however the same steps apply for SQL Server 2008

To assign a TCP/IP port number to the SQL Server Database Engine

1. Open SQL Server Configuration Manager, located in Start > Programs > Microsoft SQL Server 2005 > Configuration Tools > **SQL Server Configuration Manager**
2. In SQL Server Configuration Manager, in the console pane, expand **SQL Server 2005 Network Configuration**, expand **Protocols for <instance name>**, and then double-click **TCP/IP**.
3. In the TCP/IP Properties dialog box, on the IP Addresses tab, several IP addresses appear, in the format **IP1**, **IP2**, up to **IPAll**. E.g. one of these is for the IP address of the loopback adapter, 127.0.0.1.
4. In the **IP***All* **Properties** area box, in the **TCP Port** box, type the port number you wish this IP address to listen on (e.g. 1955 for our example), and then click **OK**.
5. In the console pane, click **SQL Server 2005 Services**.
6. In the details pane, right-click **SQL Server (**<instance name>**)** and then click **restart**, to stop and restart SQL Server.

NOTE: After the SQL Server has restarted, any client applications which connect to this SQL server should specify the server using the port number after a comma rather than specifying the instance name, e.g.

127.0.0.1\TestCaddy

Becomes

127.0.0.1,1955

Running TestCaddy with Stunnel

To connect TestCaddy to the remote SQL Server using Stunnel, you must point TestCaddy to the specified local port.

Firstly, Stunnel must be running both on client and server machines (see previous section).

If you have not yet installed TestCaddy or you have installed it but not yet run the Configuration Wizard, then go ahead and install TestCaddy on the client machine and run it from the Start menu, which will start the Configuration Wizard:

1. During the TestCaddy Configuration Wizard, at the 'SQL Instance Details' screen, set:

```
SQL Server: localhost,1956
```

Else, if you want to setup TestCaddy with Stunnel after you have installed TestCaddy:

1. Run TestCaddy

2. Go to TestCaddy Database Manager

```
File > Database Manager
```

3. At the TestCaddy Database Manager screen, change the database details

```
Click 'Change…'
```

4. At the SQL Database Connection Details screen, set:

```
SQL Server: localhost,1956
```

5. Save.

6. The list of TestCaddy databases on the SQL Server should show in the Database Manager list of databases.

7. Close the TestCaddy Database Manager screen, and then you will be prompted to select a product database.

8. Select the product database that you want to use, click OK, and continue to use TestCaddy as usual.

References

- Please read legal considerations for Stunnel as presented at http://www.stunnel.org/about/legalese.html
- We would like to recognise all those that have contributed to Stunnel as displayed at www.stunnel.org/about/credits.html
- Please read legal considerations for OpenSSL as presented at http://www.openssl.org/about/
- We would like to recognise all those that have contributed to the OpenSSL Project, as displayed at http://www.openssl.org/about/