```
+------------------------------------------------------------------------------+
| SSL and Virtual Hosts                                                        |
+------------------------------------------------------------------------------+
| Peter Thoemmes, 2011-11-10                                                   |
+------------------------------------------------------------------------------+
```

Problem with Virtual Hosts and HTTPS (SSL):

Having a virtual host environment means that on the same IP address there is more then just one web-server running.

The problem starts when setting up the encryption, because the server must select and send the SSL public key certificate before it can read the targeted server's name (fully qualified domain name, FQDN) from the HTTP request header, sent by the client. But only after it got that HTTP request it knows which server is targeted (FQDN) and so which certificate is to pick and to provide.

An extension to SSL, called Server Name Indication (SNI) addresses this issue by sending the name of the server as part of the SSL negotiation, see http://journal.paul.querna.org/articles/2005/04/24/tls-server-name-indication/:

```
    Without SNI:

    C: (SSL Handshake) Hello, I support XYZ encryption
    S: (SSL Handshake) Hi there, here is my certificate incl. encryption algo
    C: (SSL Handshake) Sounds good to me
    C: (Encrypted) HTTP Request to 'my.server.com'
    S: (Encrypted) HTTP Reply

    Using SNI:

    C: (Cleartext) I like to contact 'my.server.com'
    S: (Cleartext) BTW, I support TLS encryption
    C: (Cleartext) Lets use encryption (STARTTLS)
    C: (SSL Handshake) Hello, I support XYZ encryption
    S: (SSL Handshake) Hi there, here is my certificate incl. encryption algo
    C: (SSL Handshake) Sounds good to me
    C: (Encrypted) HTTP Request to 'my.server.com'
    S: (Encrypted) HTTP Reply
```

This enables the server to chose the correct certificate. Of course the new protocol needs to be supported by the web-browser and the web-server. According to http://en.wikipedia.org/wiki/Server_Name_Indication, the required versions are...

```
Web-browser:
    Firefox 2.0+
    Safari 2.1+
    Opera 8.0+
    IE7+ on Win 6.x+ (Vista, Win7, ...)

Web-server:
    Apache 2.2.12+ using mod_ssl
    IIS 8+
```

There is no support by IE under Windows XP, and also no support by 'wget'.
As long this matters, it is recommended to use following workaround:

Do assign an IP address to each of the virtual hosts. This can be done by IP alias definitions to the same adapter. Then base the virtual host configuration on IP addresses, rather then names. That will make the server provide the one and only certificate to an IP address, and so always the correct one.

So rather then creating name-based virtual hosts, which is easy without SSL certificates (so without SSL/HTTPS), here shown under Apache...

```
    <VirtualHost *:80>
        DocumentRoot /www/example1
        ServerName www.example1.com
        ...
    </VirtualHost>

    <VirtualHost *:80>
        DocumentRoot /www/example2
        ServerName www.example2.org
        ...
    </VirtualHost>
```

do setup an IP-based virtual hosting:

```
    <VirtualHost 172.20.30.40:443>
        DocumentRoot /www/example1
        ServerName www.example1.com
        SSLEngine on
        SSLCertificateFile /etc/ssl/certs/www.example1.com.pem
        SSLCertificateKeyFile /etc/ssl/private/www.example1.com.key
        ...
    </VirtualHost>

    <VirtualHost 172.20.30.50:443>
        DocumentRoot /www/example2
        ServerName www.example2.com
        SSLEngine on
        SSLCertificateFile /etc/ssl/certs/www.example2.com.pem
        SSLCertificateKeyFile /etc/ssl/private/www.example2.com.key
        ...
    </VirtualHost>
```

Setup the interface aliases as shown for the example above (here shown for Debian/Ubuntu Linux):

```
    # vi /etc/network/interfaces
    ...
    # Interface eth0:
    auto eth0
    iface eth0 inet static
            pre-up ifconfig eth0:0 172.20.30.50 up
            address 172.20.30.40
            netmask 255.255.0.0
            network 172.20.0.0
            broadcast 172.20.0.255
            gateway 172.20.0.254
            dns-nameservers 172.0.0.1
            dns-search hoster.com

    # Alias 0 for interface eth0:
    iface eth0:0 inet static
            address 172.20.30.50
            netmask 255.255.0.0
            network 172.20.0.0
            broadcast 172.20.0.255
            gateway 172.20.0.254
            dns-nameservers 172.0.0.1
            dns-search hoster.com
```

And finally restart networking and the web-server:

```
    # /etc/init.d/networking restart
    # /etc/init.d/apache2 restart
```

There is a special setup, where a generic name is used to reach a server:

```
                                                    +--------------------+
                     +------------+     /---------->| mymainserver.com   |
    myserver.com --->| DNS lookup |____/ ||failover +--------------------+
                     |            |      \/         +--------------------+
                     +------------+      ---------->| mybackupserver.com |
                                                    +--------------------+
```

This is to make the DNS server mapping the MAIN server's IP address to the generic URL in case everything works fine. That DNS entry is meant to be changed to point to the BACKUP server's IP address, in case the MAIN machine fails. So it is a hot backup concept, where a manual failover is done by the administrator of the DNS server.
I that case both, the MAIN and the BACKUP machine, need a second IP address to listen to:

```
    MAIN ....: 172.20.30.40 ---> alias 172.20.30.50
    BACKUP ..: 172.20.30.41 ---> alias 172.20.30.51
```

For both, the second IP addresses, the same new private key and SSL public key certificate is to be created and to be used. So both second virtual hosts use the same private key and the same SSL public key certificate (which maps the public key to the generic name 'myserver.com').

To make it work, the second virtual host just delegates incoming requests to the original virtual host (the original IP address). Here an example for the MAIN machines configuration:

```
    #
    # Generic host -> just delegating to the actual host:
    #
    <VirtualHost 172.20.30.50:443>
        DocumentRoot /www/myserver.com
        ServerName myserver.com
        SSLEngine on
        SSLCertificateFile /etc/ssl/certs/generic.crt
        SSLCertificateKeyFile /etc/ssl/private/generic.key
        RewriteEngine On
        RewriteOptions Inherit
        RewriteRule (.*) https://mymainserver.com%{REQUEST_URI}
    </VirtualHost>

    #
    # Actual host:
    #
    <VirtualHost 172.20.30.40:443>
        DocumentRoot /www/mymainserver.com
        ServerName mymainserver.com
        SSLEngine on
        SSLCertificateFile /etc/ssl/certs/mymainserver.com.pem
        SSLCertificateKeyFile /etc/ssl/private/mymainserver.com.key
        ...
    </VirtualHost>
```

APPENDIX:

Just to be complete, here the way how to create the required private key and SSL public key certificate:

To create a new private RSA key:

```
# mkdir -p /root/tmp
# cd /root/tmp/
# openssl genrsa -out generic.key 2048
```

To generate a Certificate Signing Request (CSR):

```
# openssl req -new -key generic.key -out generic.csr
Country Name (2 letter code) [AU]:US<RETURN>
State or Province Name (full name) [Some-State]:Nevada<RETURN>
Locality Name (eg, city) []:<RETURN>
Organization Name (eg, company) [Internet Widgits Pty Ltd]:<RETURN>
Organizational Unit Name (eg, section) []:<RETURN>
Common Name (eg, YOUR name) []:myserver.com<RETURN>
Email Address []:<RETURN>
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:<RETURN>
An optional company name []:<RETURN>
```

To verify the CSR:

```
# openssl req -text -noout -verify -in generic.csr
```

To create a self-signed certificate from the CSR, we provide our own private key for the signature:

```
# openssl x509 \
-req -days 365 -in generic.csr -signkey generic.key -out generic.crt
```

To verify the certificate:

```
# openssl x509 -in generic.crt -text -noout
```

To install the private key and the SSL public key certificate:

```
# mv ./generic.crt /etc/ssl/certs
# mv ./generic.key /etc/ssl/private
# chown root:root /etc/ssl/private/generic.key
# chmod 640 /etc/ssl/private/generic.key
```

This 2 files go onto the MAIN and the BAK machine!

BE CAREFUL: Never store the private key 'generic.key' unencrypted and without a secret password on a place other then the actual machine's '/etc/ssl/private' directory.