



## End to end SSL for reverse proxies

Author: Rajesh Thiagarajan and Mridul Muralidharan

Research Disclosure Database Number 501001

Published in January 2006

( Electronic Publication Date : 01 Dec 2005 11:36 )

The Research Disclosure Journal is normally published and distributed on the 10th of every month unless that date coincides with a weekend or public holiday, when it is published directly afterwards. In these cases it is always published by the 12th of every month. Every disclosure is also placed on the RD Electronic database as soon as it is received and it may be published on the database prior to being published in the next edition of Journal.

Research Disclosure is the unique international defensive publication service that allows the world's intellectual property community to establish prior art, and provides an alternative to obtaining a patent at a fraction of the cost and the time taken. It is the world's longest running, independent, industry standard prior art disclosure service.

Kenneth Mason Publications Ltd give consent for this disclosure to be printed out providing it is for personal use, or for the personal or internal use of patent examiners or specific clients only. Photocopies may be made providing it is for personal use, or for the personal or internal use of patent examiners or specific clients and not for resale and the copier pays the usual photocopying fee/s to the relevant Copyright Clearance Centre. This consent does not extend to abstracting for general distribution for advertising, or promotional purposes, for creating new collective works or for resale. This consent also does not extend to other kinds of scanning, printing or copying, such as printing, scanning or copying for general distribution for advertising, or promotional purposes, for creating new collective works or for resale. Document delivery services are expressly forbidden from scanning, printing or copying any Research Disclosure content for re-sale unless specifically licensed to do so by the publishers.

Research Disclosure Journal, ISSN 0374-4353. © Kenneth Mason Publications Ltd  
The Book Barn, Westbourne, Hants. PO10 8RS. UK  
Tel: +44 (0)1243-377977 Fax: +44 (0)1243-379136  
e-mail : info@ResearchDisclosure.com

# End to end SSL for reverse proxies

## Background:

Reverse Proxies deployed in SSL (Secure Socket Layer) mode act as a SSL VPN (Virtual Private Network) by providing security and extending the corporate network over the public internet without actually publishing any of the internal network information on the public internet's name servers. The following problem concentrates on a scenario in which SSL client certificate authentication is enabled as the part of SSL handshake both at the reverse proxy and at the destination server.

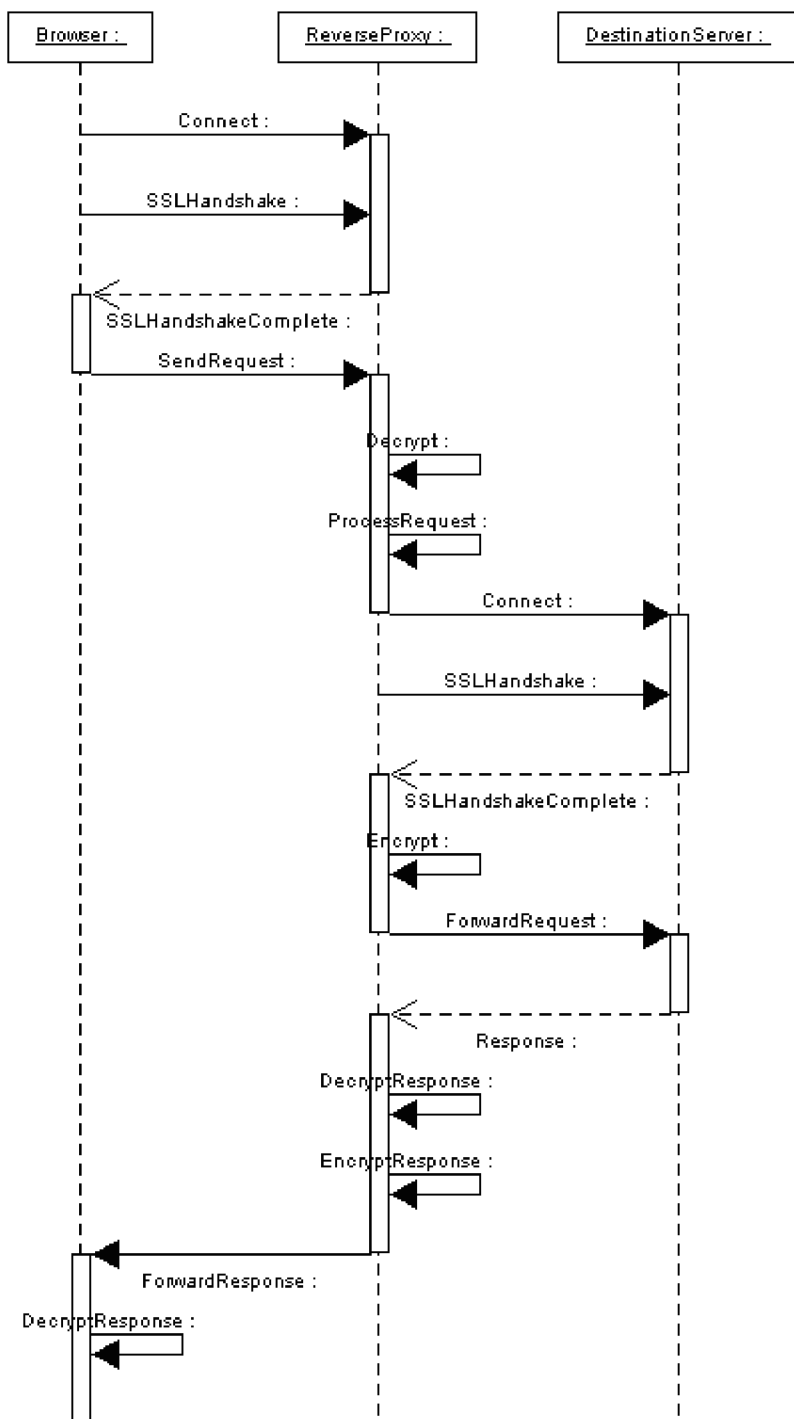
## The current solution:

The current solutions that address this problem will involve the following steps.

1. Client places a request with the reverse proxy asking it to access and fetch content from a destination server.
2. Since the reverse proxy itself requires a client certificate authentication client sends the client certificate as the part of SSL handshake.
3. The reverse proxy verifies the client certificate.
4. Upon successful validation, the reverse proxy reads the request and connects to the destination server.
5. Since the destination server is running in SSL and requires a client certificate, the reverse proxy looks up its configured certificate database for a client certificate. (It cannot use the certificate that it received from the client, as the private key for the certificate will not be available).
6. Sends the client certificate that matches the requirement of the destination as the part of SSL handshake.
7. Upon successful validation, the destination server serves the content which the reverse proxy modifies and forwards it to the client.

The high level sequence diagram (diagram 1) depicts the steps explained above, for the sake of simplicity the complete SSL handshake sequence with client certificates being sent as the part of handshake is not shown.

# Diagram 1: Current Sequence



## Issues Involved:

Summarized are some of the issues associated with such systems

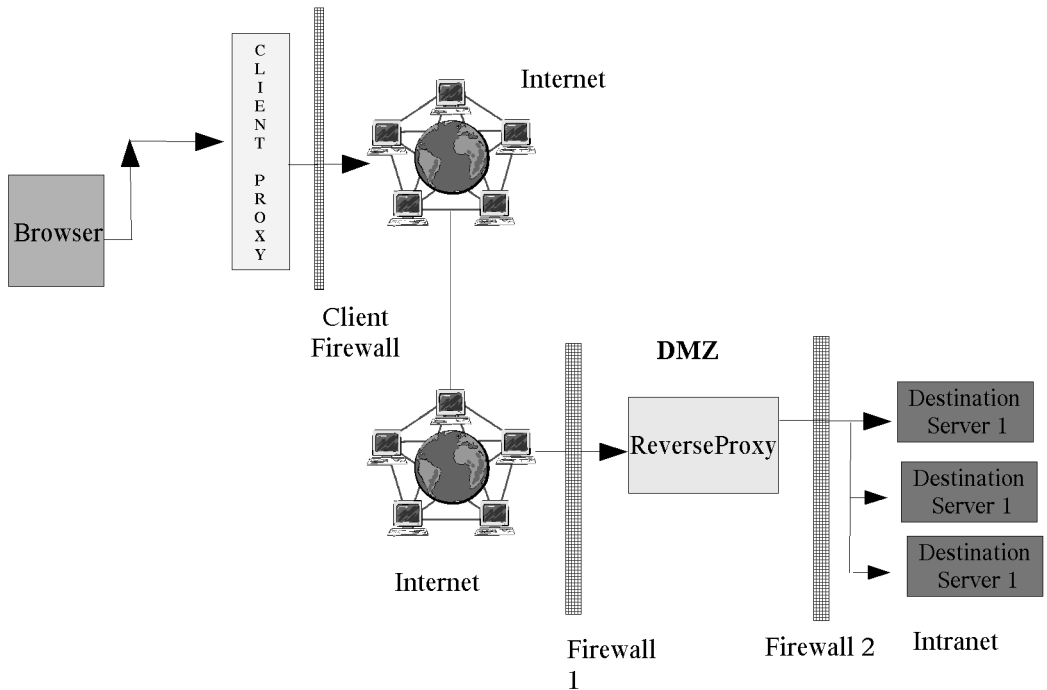
1. There is not really end-to-end SSL/TLS (Transport Layer Security) (i.e., from browser to destination server) and all the advantages that any protocol similar to security protocols like SSL/TLS provides are lost.
2. Data gets decrypted at the reverse proxy and content gets modified. So the reverse proxy becomes a potential security bottleneck and target for much of the hacking.
3. Reverse proxy needs to have the client certificate so that the destination server can trust the client. Storing client certificate at a remote server is not secure. Alternatively, some form of common proxy certificate gets installed on the reverse proxy which it uses as the client certificate for all connections, which is also not a secure option.
4. Encryption/Decryption happens at the reverse proxy which causes performance degradation.
5. There is no data integrity between the end server and the client.
6. It is not the actual client certificate that is getting presented to the destination server i.e., the trust between the actual client and destinations servers is established by means of a certificate proxy, which itself is a security issue.

## Summary of presented idea:

This idea works on the principle of automatically upgrading and downgrading from/to an SSLSocket to PlainSocket. Also the solution outlines few concept called “Sequential Connect” and “Double SSL handshake” which are explained in detail in later sections.

The setup required to describe this idea is as follows. Diagram 2 depicts the setup required.

## Diagram 2: Setup



The Reverse Proxy typically resides in the DMZ (Demilitarized Zone) and its IP address gets published in the public(internet) DNS (Domain Name System) - perhaps it is the only IP that gets published. The client is protected by a firewall and accesses the corporate intranet information via a client proxy that connects to the public network (internet).

The basic idea is as described below :

The request to connect to a SSL enabled destination server will reach the client proxy from the browser. The client proxy will need to go through the reverse proxy to reach the destination server.

This client proxy (CP) will have a SSL Socket connection to the reverse proxy. After the successful SSL handshake and subsequent reverse proxy validation , the reverse proxy will make the requested connection to the destination server at the specified port.

Now , both the client proxy and reverse proxy will downgrade their connection from SSL to plain and client proxy will respond to the browser with a response indicating successful connection. The browser will now continue with its handshake directly with the destination server.

Hence, there are two SSL handshakes happening here :

1. Between the client proxy and reverse proxy – hence security validation and authorization checks at the reverse proxy can be done.
2. Between Browser and Destination Server (directly) – hence the browser can directly talk to the destination server.

### **Description:**

The following is the sequence of states that the entire setup goes through to overcome the issues summarized in the background section.

(Also assume that the client proxy is modified in such a way it supports the following message sequence. As the proposal unveils these modifications are not really required at the client proxy or any intermediaries.)

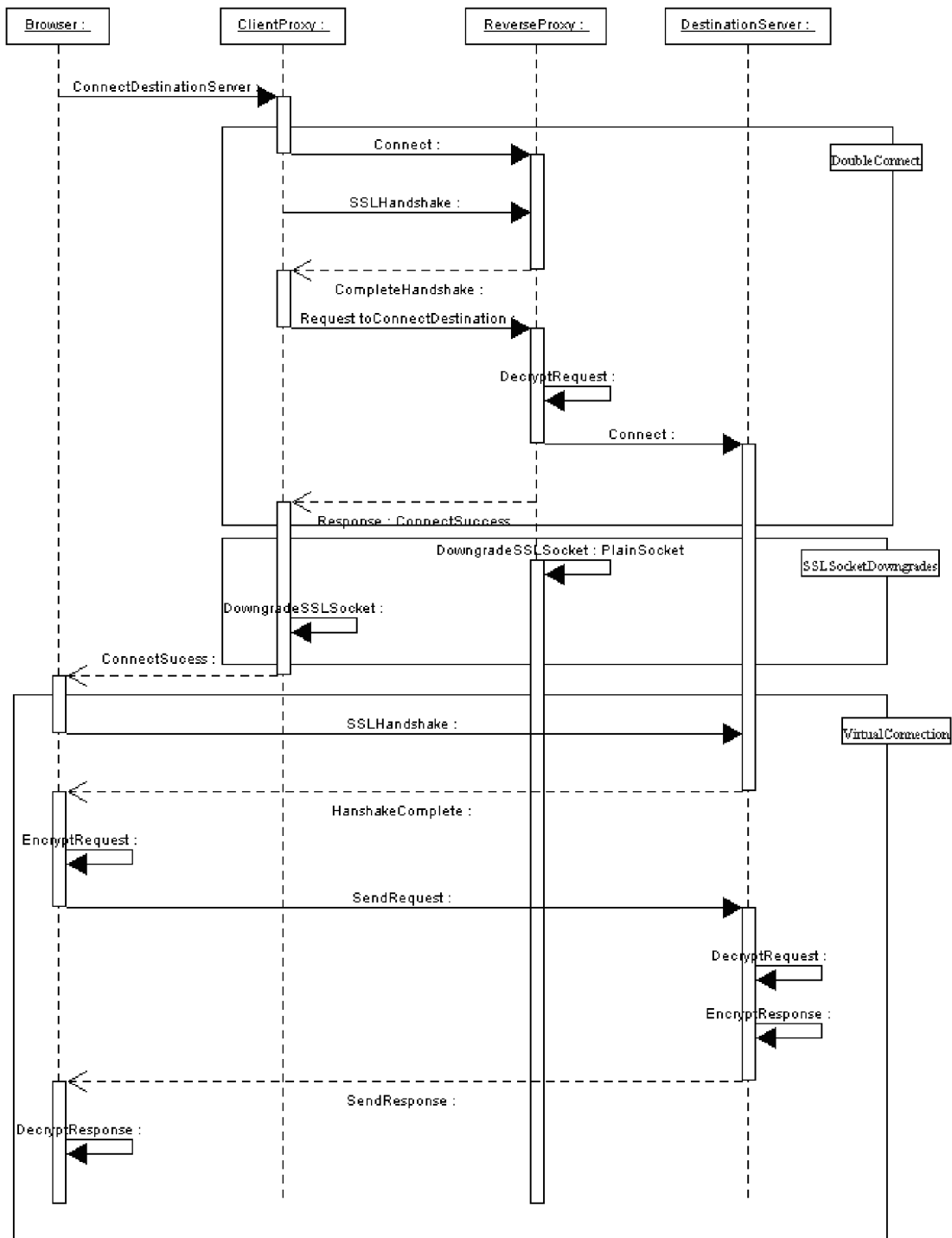
1. Browser places a request to the destination server which is running in SSL. Since it cannot reach the destination server directly it will place a request to the client proxy to connect to the destination server. Typically on HTTP (Hyper Text Transfer Protocol) it would be a CONNECT (as described in HTTP RFC) command to the client proxy(CP).
2. The destination server is protected by a reverse proxy which itself is running in SSL and requires a client certificate as the part of SSL handshake. The client proxy would connect to the reverse proxy instead of the actual destination server.
3. Upon connecting, the client proxy looks up it's certificate database and searches for a client certificate which will be accepted by the reverse proxy and uses it to do the SSL handshake with the reverse proxy.
4. Upon validating the client certificate the reverse proxy accepts the connection and waits for requests to process.

5. Once the SSL handshake completes successfully the client proxy places a request (i.e. a CONNECT request) to the reverse proxy asking it to connect to the destination server which the client actually requested for. The whole above sequence is referred to as “Sequential Connect”.
6. The reverse proxy connects to the destination server and acts as a simple proxy server. It would not do any SSL handshake with the destination server. Instead it creates a tunnel between the client and the destination server so that the SSL handshake can happen between the client and the end destination server.
7. Upon successful connection establishment the reverse proxy sends an acknowledgment to the client proxy stating that connection establishment is done.
8. Client proxy upon receiving this acknowledgment sends a success status code in response to the initial request (CONNECT command) that it received on step 1 from the browser.
9. Before any data is transmitted between the client proxy and reverse proxy, both these proxies downgrade their SSL Socket to plain socket to avoid double encryption – as the connection until then was SSL between the client proxy and reverse proxy. Now the browser would initiate another SSL connection over this between itself and the destination server.
10. Once this is done, all the data that browser transmitted are tunneled to the destination server. So the browser does a SSL Handshake with the destination server based on the tunnel set by the client proxy and the reverse proxy. This enables the actual client certificated to be tunneled to the destination server.

Diagram 3 depicts the steps explained below. For the sake of simplicity, the complete SSL handshake sequence with client certificates being sent as the part of handshake is not shown.

There are three different areas that are highlighted in Diagram 3, namely “Sequential Connect”, “SSL Socket Downgrade” and the final “Virtual Connection” that's established between the client and the destination server. These are the fundamental concepts upon which this proposal is based on.

# Diagram 3: Modified Sequence/Proposal



Please note the entire traffic is in SSL. There is no custom encryption or non-standard security protocol used anywhere in the system.

The above ensured that there is end-to-end SSL encryption in a reverse proxy setup without actually publishing the end servers IP on the public internet DNS. However it did not solve the issue of needing a client certificate in some remote location such as the client proxies certificate database. Also, it required few modifications in the client proxy to do SSL handshake and to know which hosts required connecting to the reverse proxy. These issues can be solved as below.

1. The client proxy may not mean a proxy that is physically sitting on a separate system. Instead it can be deployed on the same system as the client browser. Also it can share the same certificate database that the browser actually is using. In this case there can be some mapping that gets installed along with this proxy which specifies for which hosts it should connect to the reverse proxy.
2. The client proxy can be a downloadable code such as a Java Applet or an ActiveX control that gets automatically downloaded on the first request from the browser to the reverse proxy. This code can modify the browser's proxy settings accordingly to point to itself.
3. As the part of this downloaded code a temporary certificate may be issued for this applet to connect to the reverse proxy while only the destination servers certificate is stored in the browsers certificate database.
4. In both the previous cases (2 and 3) the mapping required for routing request can be downloaded along with the code. This mapping specifies which hosts require connecting via the reverse proxy.

The previous solution also solves a related problem which comes up when the user is accessing multiple internal sites differing in domain names that are protected by the reverse proxy and which require client certificate for authentication. As said above, both reverse proxy and the destination server requires client certificate for access. The reverse proxy and destination server may not be on the same domain. In such case both may require client certificates to authenticate the user. It's not always necessary that both these certificates are the same. In case they are different it adds additional layer of security, i.e., two different client certificates being presented to two different servers for authentication. This is referred to as as “Double SSL Client Certificate Authentication.”

## Advantages:

1. There is End-to-End SSL from the actual client to the destination server.
2. No decryption or encryption of the traffic in the intermediaries.
3. Direct trust establishment between the destination server and the actual client.
4. Client certificate is not stored in any remote location except on the clients machine.
5. Data integrity and all advantages that SSL provides are not lost.
6. There is no need to open any additional ports in the corporate firewall that protects the intranet.
7. There is no need to modify any of the intermediaries network elements such as proxies, router etc.

The entire proposal is based on downgrading and upgrading a SSL Socket, and this can be a tricky issue as most of SSL implementations do not support it for security reasons. Achieving the same functionality using some non-standard encryption techniques especially between the client proxy and reverse proxy and downgrading such to plain socket is much easier. However the above proposal attempts to solve the problem using only SSL so that any intermediaries that checks for SSL specific packet do not drop any of the data that are passing through them. In fact, having this solution adhere to standard SSL specification is one of the primary claim that this proposal is attempting. This is absolutely necessary to make sure that an intermediaries do not drop packets because the traffic does not comply with the SSL specification.

Most of the SSL implementations do not support downgrading a SSL socket to a plain socket but they support the reverse. The following description defines how this downgrading of SSL Sockets can be done even though the underlying SSL layer does not support it.

At the Reverse Proxy side (an example of downgrading a SSL socket to plain socket):

1. The reverse proxy would listen on plain Server Socket. This will be the standard port of the reverse proxy. A client connection will return a plain (clear text) socket to the reverse proxy.
2. The reverse proxy would also create a SSLServerSocket bound to 127.0.0.1 on the reverse proxy machine. This way, only processes from the same system can connect to this socket.

3. When a client connects to the plain socket (from 1), a standard pipe would be created between the plain socket that the reverse proxy obtained as the result of client connect and a socket obtained from the SSL Server Socket.
4. This allows it to read and write data from both the plain socket and SSL socket. This way, reverse proxy can use the SSL socket to send/receive encrypted data while use the plain socket for reading raw (clear text) data.
5. By doing the previous step for every connect from a client there would be 2 sockets associated with it, A plain socket and a SSL socket. Hence, using this mechanism enables that the system to upgrade or downgrade a plain socket to SSL socket and vice-versa.