Persisting SSL Connections

Deb Allen, 2008-05-08

Many customers use LTM to handle SSL encrypted traffic, and traffic that requires SSL certificate authentication and encryption often also requires persistence to a specific server for the life of an application session. LTM is capable of meeting most security requirements for traffic encryption with the 3 most common high-level SSL configurations: *SSL Offloading*, *SSL Re-encryption*, and *SSL Pass-through*. The available persistence options vary depending on which SSL configuration is implemented. In this article, I'll briefly describe each mode, and the persistence options available for each.

SSL Offloading

LTM is offloading SSL (decrypting SSL and using a cleartext connection to the real server) if you have only a clientssl profile configured on your virtual server. This configuration is the recommended option if your application requires persistence and cleartext between LTM and the servers is acceptable, since it is most optimal and offers the most flexibility as far as persistence is concerned. SSL offloading is most optimal because it allows LTM to do the heavy lifting of encryption on the client side while completely eliminating any overhead of encryption on the server side. At the same time, it's most flexible regarding persistence options: All of the persistence options available for unencrypted traffic are available when LTM decrypts the conversation:

Source Address: Also known as simple persistence, source address affinity directs requests to the same server based solely on the source IP address of a packet.

Destination Address: Also known as sticky persistence, destination address affinity directs session requests to the same server based solely on the destination IP address of a packet.

Cookies (for HTTP only): Cookie persistence uses an HTTP cookie stored on a client's computer to allow the client to reconnect to the same server previously visited.

Hash: Hash persistence allows you use an iRule to create a persistence hash based on any persistent request data.

MSRDP: Microsoft Remote Desktop Protocol (MSRDP) persistence tracks sessions between clients and servers running the Microsoft® Remote Desktop Protocol (RDP) service.

SIP: Session Initiation Protocol (SIP) persistence uses th SIP CallID to track the servers to which messages beloging to the same session are sent. (SIP is a protocol that enables real-time messaging, voice, data, and video.)

SSL: SSL persistence is persistence option specifically intended for use with non-terminated SSL sessions, and tracks the server to which connectins shoud be sent using the SSL session ID.

Universal: Universal persistence allows you to write an iRule expression that defines what to persist on in a request, and can use nearly any persistent request information to track sessions: Protocol headers, HTTP cookies, URI parameters, session IDs in the data stream, etc.

The most protocol or application-specific persistence option available is recommended. (It's worth noting that whatever persistence option is optimal for the unencrypted version of your application should also be optimal when offloading SSL to LTM.) For HTTP applications, some form of Cookie persistence is our most common recommendation, with Simple or Universal persistence as options if cookies are not supported by the expected client base. You may have noticed that SSL persistence didn't make the list. In fact, it isn't recommended unless it's the only available option, for reasons explained below in the section about SSL Pass-through.

SSL Re-encryption

LTM is re-encrypting SSL (decrypting SSL and re-encrypting over the connection to the real server) if you have both a clientssl and serverssl profile configured on your virtual server. This configuration is the recommended option if your application requires persistence on session data but must also be encrypted between LTM and the servers. An optimized SSL handshake and intelligent keep-alives for connections with the real servers still allows LTM to lighten the load on the servers even though they still have to perform encryption/decryption tasks.

As with SSL offloading, all of the persistence options available for unencrypted traffic are available when LTM decrypts the conversation, and the most protocol or application-specific persistence option available is recommended.

SSL Pass-through

LTM is performing SSL pass-through (neither decrypting nor re-encrypting SSL, instead forwarding the SSL handshake and connection directly to the real server) if you have neither a clientssl or serverssl profile configured on your SSL virtual server. This configuration is the recommended option only if your application cannot tolerate SSL proxying or decryption is not an option.

For SSL Pass-through configurations, the persistence options are severely limited: Since LTM is not decrypting the conversation, only the non-SSL-encrypted information in the session is available for use as a session identifier. The primary pieces of persistent unencrypted information in an encrypted SSL flow are the source and destination IP addresses, and the SSL session ID itself, so only Source Address, Destination Address, or SSL persistence will work with SSL Pass-through configurations.

Our recommendation, as with SSL offloading or re-encryption, is still to choose persistent token data closest to the application, so in this case, SSL is the preferred persistence method for SSL Pass-through.

SSL persistence is intended to track non-terminated SSL sessions using the SSL session ID. Using this lower level data rather than actual application session identifiers such as sessionIDs or cookies is less reliable since SSL IDs are subject to re-negotiation or re-use during the course of an application session outside the application's control or awareness. However, it's the best information available, so we recommend setting SSL persistence as the primary persistence method, then set Source Address as a backup persistence method to stick new connections to the same server even if the SSL session ID changes mid-application session. (Note: Users behind large mega-proxies such as AOL may move from one proxy to another during the same application session, thus changing their source IP address to another within a very large address block. If your application will be serving users behind a large megaproxy, be sure to set the persistence mask for Source Address persistence to encompass the entire range of possible alternate addresses.)

Get the Flash Player to see this player.
20080805-PersistingSSLConnections.mp3

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.	F5 Networks	F5 Networks Ltd.	F5 Networks
Corporate Headquarters	Asia-Pacific	Europe/Middle-East/Africa	Japan K.K.
info@f5.com	apacinfo@f5.com	emeainfo@f5.com	f5j-info@f5.com