

# Web HTTP Connection Patterns When Load-Testing Using LoadRunner

**HTTP Connections** In a web environment clients communicate with the server using HTTP connections. A connection can be one-shot or persistent. One shot connections are used for one interaction, when the server has responded the connection is closed. Persistent connections are kept open for use in subsequent interactions. Persistent connections are closed by the client when it has finished with the server or by the server if the connection is inactive for too long.

**Connection Patterns** Load-Testing aims to simulate operating conditions to allow assessment of the system's behaviour. An effective simulation must have the key characteristics of the scenario being simulated. One such characteristic is the pattern of HTTP connections.

The connection pattern is characterised by the rates of connection creation and closure events and by the number of concurrent connections. These can have a major impact on how many tasks the server is required to perform and on how much processing each task requires. They can also make the server run out of resources and fail.

The pattern emerges from the behaviour of the clients and the behaviour of the server. The clients may request one shot or persistent connections and may attempt to use a number of connections in parallel. The server determines whether persistent connections are supported and the timeout to apply to connections. The arrival rate influences the rate that connections are created and closed and, in conjunction with the duration of existing connections, the number of concurrent connections that the server must deal with.

The average duration of a connection, through its effect on the number of concurrent connections, can be highly significant. For one-shot connections it is determined by the time taken by the server to deal with the request and the delays arising from network transit time. Where persistent connections are in use the client processing times and operator response times, simulated as Think-Time, influence the connection duration.

**Simulation Accuracy** The connection pattern generated needs to be right. It should match the pattern that the situation being simulated would produce. This involves:

- (a) Ensuring the server is configured correctly with respect to persistent connections and timeouts<sup>1</sup>.
- (b) Reflecting the characteristics of the production network in the test system.
- (c) Appropriate simulation of client behaviour.

**Simulation Design** The design should reflect the scenario to be simulated. The production network characteristics

together with the type of connections and the potential concurrency must be addressed. Care should be taken when accelerating client actions, by reducing Think-Time, due to the impact on average connection duration<sup>2</sup>.

**Implementation in LoadRunner** The simulation is implemented using LoadRunner. Vuser bandwidth restrictions and Wan emulation allow a first level emulation of the production network. The Vuser script and run-time settings define client behaviour. The main controls are:

- (a) The Keep-Alive HTTP connections setting. This sets the default request *Connection* header and determines whether the Vuser closes the connection on receipt of the server response.
- (b) Direct manipulation of the request *Connection* header using the *web\_...\_auto\_header()* functions. These can override the default request *Connection* header, however the Vuser will still close the connection if Keep-Alive is disabled.
- (c) Limit on concurrent connections from the Vuser to any single server this is set using *web\_set\_connections\_limit()* or the *web\_set\_socket\_option()* function.
- (d) Limit on total concurrent connections from the Vuser to all servers this is set using the *web\_set\_socket\_option()* function.
- (e) Explicit closure of persistent connections using the *web\_set\_socket\_option()* function.

**Watch Out For** When persistent connections are in use, which is the norm, and iterations are paced, Vusers do not close the connections until the start of the next iteration. This leaves connections open when a real client would have closed them. When a client would disconnect from the server connections should be closed using *web\_set\_socket\_option()*.

**Monitoring at RunTime** It is easy to make a mistake and to run a simulation where the connections do not reflect the scenario. This can invalidate the Load-Testing. During execution connection events and the number of connections should be monitored to detect anomalies. For example on MS IIS the *Web Service* counters *Connection Attempts/sec*, *Current Connections*, *Maximum Connections* and *Total Connection Attempts* indicate connection behaviour.

1. If the server configuration is not defined by the system then testing should address varying configurations and include sensitivity analysis on the timeout setting.
2. For a more detailed discussion see our white paper reference STTN-2004-01.

**sqc**

[www.sqc.co.uk](http://www.sqc.co.uk)

SQC Technology Ltd.  
Coton Park House, Linton, Swadlincote,  
DE12 6RA, UK.

SQC Technology provides software testing, software test management and software test automation services and consultancy. This includes formulating software test strategies, test analysis and development of test suites, test automation and interim management. More information on our advanced capabilities and innovative approaches can be found on the company web-site. [www.sqc.co.uk](http://www.sqc.co.uk)