

HP LoadRunner software best practices guide: validate and optimize the performance of Citrix Presentation Server



Table of contents

Introduction	3
How does Citrix Presentation Server work?	3
Performance Validation and Optimization	3
Performance Validation and Optimization lifecycle for Citrix Presentation Server	4
Planning an effective load test for Citrix Presentation Server 4.5 using HP LoadRunner 9.0 software	5
How does HP LoadRunner work with Citrix Presentation Server?	5
High level conceptual goals	6
Identifying the business process to performance test	6
Capturing user location distribution versus business process	7
Documenting user steps and data	8
Transactions for user response time	8
Documenting statistics for overall performance optimization	9
Best practices for scripting on Citrix Presentation Server 4.5 with HP LoadRunner 9.0 software	10
Scripting with Citrix Agent versus without Agent	10
Client and server setup for Citrix	10
Load generator and VUGen setup for Citrix	10
Citrix Presentation Server configuration for load testing	10
Independent Computing Architecture (ICA) files	11
Synchronization	12
What is synchronization?	12
Adding synchronization during recording	12
Adding synchronization after recording	13
Adding logic to the synchronization	15
Wildcard	16
Best practices for Citrix script development	16



Table of contents (continued)

Recording options	16
Using the keyboard versus a mouse	18
Window resizing	19
Display settings and color resolution	19
Recording a complete session (running a clean session)	20
Using various Action sections	20
Timeout settings	20
Continue on error setting	21
Replay tips and tricks	22
Setting Think Time	22
Generate snapshot on error	23
Graphics device interface (GDI) resources	23
Debugging tips and tricks	24
Logging	24
Show client during replay from Controller	24
Best practices for scenario execution of Citrix Presentation Server 4.5 virtual users with HP LoadRunner 9.0 software	26
Load generator considerations	26
Display settings	26
Location	26
Concurrency	26
Ramp rates	26
Logging and viewable Citrix clients	27
Debugging	27
Viewing Citrix clients	27

Introduction

Today's IT environments contain a proliferation of computing endpoints and application types. The increasing adoption of web applications, software as a service and hosted applications is adding significant complexity to existing legacy, client-server and desktop application environments. IT must continue to support all application environments and user scenarios, both new and old. To address today's dynamic application delivery challenges, IT needs a new integrated and innovative approach to application delivery and application performance testing. Citrix Presentation Server and HP LoadRunner software provide this new approach—an approach that streamlines application delivery for all Microsoft® Windows® applications, mitigates risk and simplifies application performance validation—enabling IT to meet today's stringent business requirements.

How does Citrix Presentation Server work?

In a traditional environment application thick-clients are installed locally on a client machine or accessed via a browser running on a client machine. Communications take place directly over a network between the client on the users' machine and application servers located in a data center. Increasingly, data centers are located in remote locations relative to the end-user. This traditional model requires significant network bandwidth, and often impairs user productivity with slow application response times. This traditional model also has potential security issues as business and application data is sent across the wire and then resides on the local desktop or laptop.

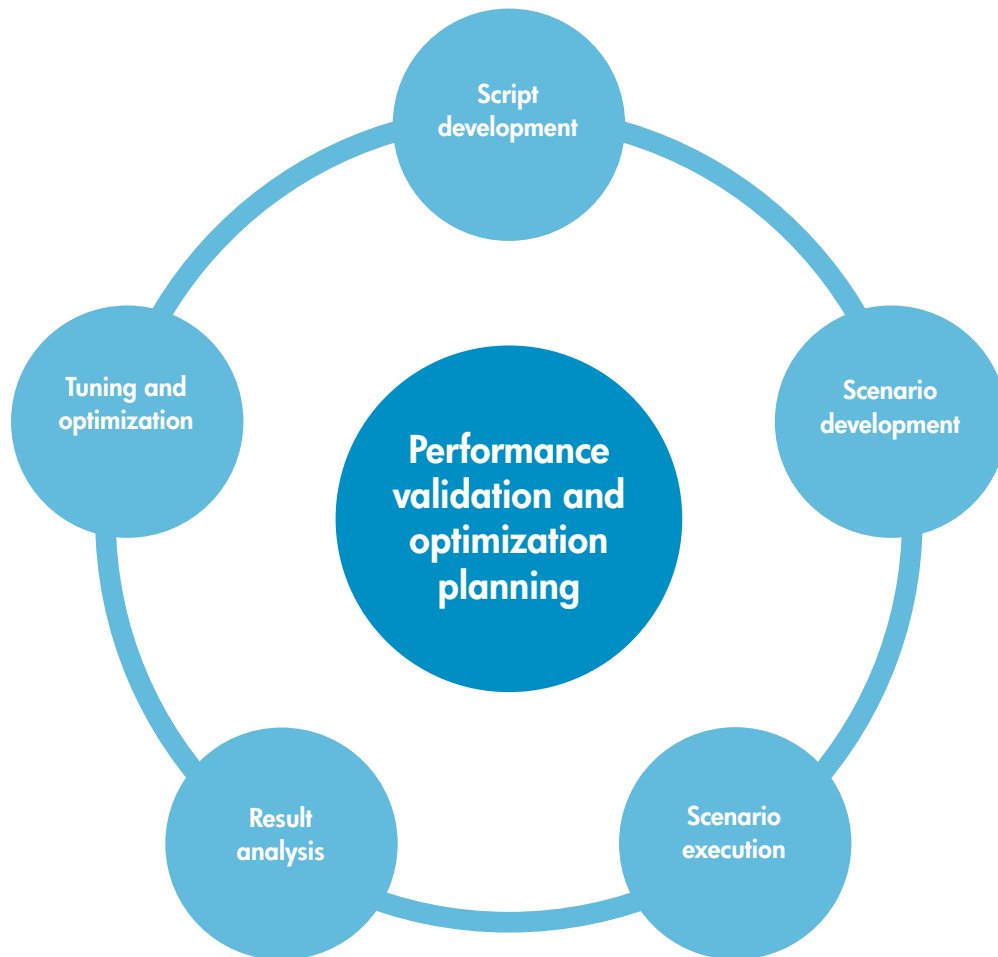
Citrix Presentation Server solves these problems by centralizing applications and data in secure data centers. Presentation Server virtualizes applications by separating where the application is used from where it runs. All components of applications are installed, maintained and secured in the data center, while only screen displays, keyboard entry and mouse movements traverse the network. IT consolidates application management centrally, while enabling users the freedom to operate from any location, on any device, over any network. Citrix Presentation Server is a complete application delivery system for Microsoft Windows applications that offers both application virtualization and application streaming to enable the best access experience for any user, with any device, working over any network. Presentation Server's underlying intelligence automatically selects the best delivery method for the user, the application and the location.

Performance validation and optimization

Like all application environments, Citrix Presentation Server implementations can benefit from performance tuning in order to be configured effectively to meet expected loads with optimal performance. HP Software and Citrix have joined together to provide a performance validation and optimization solution that leverages key capabilities of HP LoadRunner 9.0 software for conducting performance testing in Citrix Presentation Server 4.5 environments. HP LoadRunner is an industry-standard load testing and performance validation solution that helps in predicting system behavior and isolating potential performance bottlenecks. HP LoadRunner enables IT to optimize the performance of Citrix Presentation Server before going into production.

This white paper provides information related to the performance validation and optimization lifecycle of Citrix Presentation Server 4.5 environments. We will discuss all aspects of performance testing including best practices for planning, setup, script development, scenario execution and result analysis unique to Citrix application delivery infrastructure using HP LoadRunner 9.0 software.

Performance validation and optimization lifecycle for Citrix Presentation Server



Planning an effective load test for Citrix Presentation Server 4.5 using HP LoadRunner 9.0 software

Planning is critical to all load testing projects. It is especially important during Citrix implementations due to the complex nature of the interaction between HP LoadRunner and Citrix. Let's begin with a basic understanding of how HP LoadRunner works with Citrix.

How does HP LoadRunner work with Citrix Presentation Server?

HP LoadRunner interacts with the application using virtual users (Vusers) to emulate real-life conditions; this is achieved by recording the communication between the user and the server. Therefore, every recording can be different depending on the environment (i.e. web servers, application server, database server or the presentation server). If we take an example of a standard web server, the client communicates with the server using HTTP calls. These calls are generated to the server embedded with the data that is needed to process a request. When the web server receives the request from the client, it takes the appropriate action by either processing the request to search, update, etc. When the request is processed by the server, a response is sent back to the client with an appropriate result for the original request.

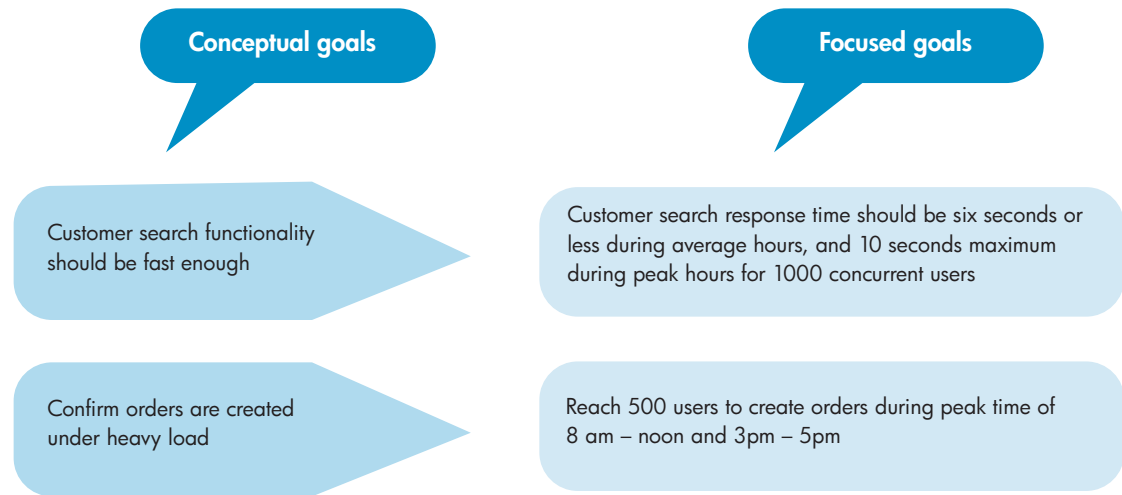
Citrix Presentation Server delivers applications to users via Independent Computing Architecture (ICA), a communication protocol developed by Citrix by which servers and client devices exchange data, and which separates an application's logic from its user interface. The ICA protocol encrypts and transports an application's screens from the server it is running onto the user's client device, and returns the user's input to the application on the server. As an application runs on a server, Presentation Server intercepts the application's display data and uses the ICA protocol to send this data to the ICA client software running on the user's device, increasing application performance.

When HP LoadRunner records a Citrix script, it captures all key strokes, mouse clicks and other real interaction along with the screen coordinates. The objects within the Citrix session are recorded in a form of bitmap in the HP LoadRunner script. Citrix scripts also require synchronization between the client and application to ensure that the user is not ahead of the actual request. For example, if the user is searching for an item they type in the item number and press enter. This generates a request to the server. Before the user can move on to the next step of the business process, however, they have to ensure that the appropriate search results were returned from the server. Otherwise, the script will fail since the next step is not visible on the screen. This requires synchronization to make sure that the expected results were returned from the server. Synchronization also helps when running multiple users and the application under test is stressed, thereby yielding slow response. It ensures that the client and the server are always in sync.

High level conceptual goals

Performance validation and optimization should always be started with high level conceptual goals. These goals start with the understanding of what is expected from the application in the production environment with actual users using the application. Conceptual goals help in determining more focused goals on a business process or transaction level. Focused goals are translated into the actual business scenario, and then compared to the results to see if the expectations were successfully met.

Examples for the conceptual goals can be as follows:



Identifying the business process to performance test

The critical part of planning the load test is to identify the business process or scenario to be tested that emulates a realistic model of the production environment. This does not mean that all business processes or permutations of the scenario should be included in the load test. The goal should be to take the triage approach instead of the operational model to generate 80 percent of the load from 20 percent of the business processes or scenarios. The triage approach is based on three basic criteria listed below.

Criteria	Description
High usage	Transactions that are used by a high number of users; for example, "Search Customers" is used by 1000 users compared to "Print Invoices" which is used by five users in the back office simply printing invoices upon request.
High volume	Business processes that generate a high number of transactions in terms of the total volume; the "Create Order" transaction creates 10,000 orders per day for each site.
High risk	These transactions are critical to the business and can have an adverse impact on day-to-day operations. When the shopping cart functionality of the eCommerce website is unavailable or slow in response, it can have a direct effect on the business and therefore the transaction is classified as a risk.

The triage approach helps in isolating important business process that should be included in the load test. Often this information is gathered from various sources such as online production statistics, business teams, experienced users, developers, system administrators and database administrators. The key is to assemble this information in an organized form so that it can help determine what transactions should be included in the scope of the project. There are many ways to organize the vast amount of planning information collected. The "Performance validation scoping worksheet" illustrated below encapsulates the information gathering in a very efficient manner.

Performance validation scoping worksheet

Business process name	Business risk (low, medium, high)	Transaction complexity (low, medium or high)	Typical number of users (per hour)	Peak number of users (per hour)	Typical number of transactions (per hour)	Peak number of transactions (per hour)
------------------------------	-----------------------------------	--	------------------------------------	---------------------------------	---	--

Capturing user location distribution versus business process

Once the final list of business processes has been assembled using the performance validation scoping worksheet and information is gathered from the appropriate resources, the next step is to determine the user's location distribution for each business process. This is critical if the user base is segregated across various locations with each user base having its own critical function to perform within the application. As an example, an organization can have [Site 1] as a "Sales/Distribution" center and [Site 2] as a "Manufacturing" center. These two locations serve in their own capacity to fulfill the business function, and use the application accordingly. [Site 1] will perform sales functions like creating sales orders, thus the business process used by sales team members will greatly differentiate from someone using the application from the manufacturing group. Documenting this type of user distribution information versus business process helps in generating appropriate load from each site with the related business process for their function in the organization. This information can be easily captured using the "User location distribution versus business process" template illustrated below.

User location distribution versus business process

Business process	Boston	New York	Tokyo	London
Create order		15	10	30
Create delivery list		5	3	8
Edit formula	10			
Process plant order	5			
Create packaging	5			

Documenting user steps and data

Documenting user steps is an important part of planning. The goal at this stage is to capture actual user behavior, and the flow of steps that need to be emulated for each business process. It is critical that the business process steps resemble how the user will perform these functions in real life so that when the load test is executed, it closely reflects the load that is generated on the application by the actual user in production. Experienced users and/or subject matter experts (SME) are key resources that can provide significant input related to the user steps. It is also a good practice to use screen recording utilities, such as SnagIt, to capture the user interaction with the application. This is especially useful when an experienced user or the SME is walking through the business process. The screen recording can be replayed back at a later stage to revisit, or understand the business process flow, or to look at the data used. Additionally, it is also a good practice to document the user steps, and the data input values as part of the planning phase. This can be achieved by following the user steps template illustrated below.

Business process step-through click stream		
Step	Action type	Step details
*	User	Launch Citrix ICA client
*	Transaction	Start [login] transaction
*	User	Enter user ID and password
*	Transaction	End [login] transaction
1	Transaction	Start click [Create Order] transactions
2	User	Click Create Order link
3	Transaction	End click [Create Order] transactions

Data is an integral part of load testing an application under test—capturing the data values needed for the business process is extremely vital to the success of the load testing effort. The data requirements for each business process should be documented with information such as name of the value, type of the value and the source. This can be easily attained by using the following template for each business process.

Data and parameterization details			
#	Parameter name	Description	Data source
1	User ID	Login user name	Contact DBA
2	Order No.	Order number for search	Create order
	...		

Transactions for user response time

HP LoadRunner transactions are inserted in the script to capture the response time for the specific action of the user for a step or group of steps. During the execution, the transaction is started, followed by a user step(s), and closed to determine the time it took to complete each step(s). The response time captured during the execution is typically compared to the service level agreements (SLA) established during the planning phase. It is important to note that the transactions and their respective SLAs should be documented before any script development or load test execution is started to avoid any misrepresentations. This can be easily documented by using the following template.

Transactions for user response time and service level agreement		
Transaction name	Description	Threshold (SLA)
Login	Login user to the application (sample)	<= 4 > 8 > 12
Click Order link	User clicks on the link to begin Create Order	<= 4 > 8 > 12
Save order	User clicks on the Save button to save the order	<= 5 > 9 > 12

Documenting statistics for overall performance optimization

Performance optimization goals should be documented during the planning stage. This requires a team effort; therefore it is important to engage key stakeholders to decide on a common goal for the statistical performance optimization to be achieved. This process begins with determining the load that should be generated on the application under test—like peak concurrent number of users, average volume per hour, and the total execution time for the scenario. Service level agreements, or the transaction response time threshold established for the key transactions by the business user, or SME should be included to demonstrate the correlation between the user load, volume and response time. Documenting and establishing the threshold for the system resources utilization—such as CPU utilization, disk I/O, paging, memory consumption and network latency—is extremely essential. These help in comparing the metrics captured during the test execution to the expected performance threshold determined early in the project phase. The “Goals for application performance optimization” template presented below will help in documenting these thresholds and SLAs for load testing projects.

Goals for application performance optimization

Workload	Example	Target
Peak concurrent user load	250 users	# users
Peak transaction rate	1000 trans/hour	# trans#/hour
Duration of peak load	2 hours	# hours

End-user response times (90%)	Example	Target
Transaction#1 Threshold	6 secs	# secs
Transaction#2 Threshold	6 secs	# secs

Errors	Example	Target
Error tolerance	1 % (of errors)	# % (of errors)
Transaction failure tolerance	1 % (failed) transaction)	# % (failed) transaction)

Resource utilization	Example	Target
Network bandwidth threshold	40 Mbps	# Mbps
CPU threshold	80 %	# %
Memory consumption	90 %	# %

Best practices for scripting on Citrix Presentation Server 4.5 with HP LoadRunner 9.0 software

Scripting with Citrix Agent versus without Agent

HP LoadRunner 9.0 software allows script development with or without the Citrix Agent. There are many benefits to using the Citrix Agent since it was developed as an add-on to the existing HP LoadRunner protocol to provide the following benefits:

- Ease of scripting by making the script more readable and intuitive. This is attained by adding object names to the new function library developed for the Citrix protocol that is used with the Citrix Agent, i.e. `ctrx_obj_mouse_click()`
- Provides detailed information on all objects recorded by using object names along with X,Y coordinates, therefore making it easier for HP LoadRunner to find objects during replay
- Provides additional functionality for the “Right Click–Expand” option, allowing the script developer to use the Tree view snapshot to add synchronization, text retrieval and verification functions
- Allows the visual display of the object within the snapshot to see what was selected during the recording, which is known as the “Active Object Recognition” functionality

The benefits of using the Citrix Agent make it an obvious choice for any Citrix load testing effort as it reduces script development time, provides more visibility into the objects, and offers additional functionality to ease the after recording process for scripts. This white paper will now focus on examples and solutions that are available with the Citrix Agent.¹

Client and server setup for Citrix

Load generator and Virtual User Generator (VUGen) setup for Citrix

Citrix ICA client should be installed on the machine that is used for script development and on all the load generators included in the load testing effort. If the Citrix client is not installed, then the VUGen machine will not launch the ICA client, and similarly the script will fail on the load generator upon execution. The Citrix client can be downloaded from the Citrix website at: <http://www.citrix.com/clients>.

Citrix Presentation Server configuration for load testing

Citrix Presentation Server must be configured to close the session when inactive or because of an unexpected break in connectivity from the client. Terminating the session completely will ensure that a new clean session is started each time the user fails or tries to reconnect to the server. If the setting to terminate the connection is not enabled, the user will connect to the last session and be placed in the old workspace. This setting is critical to successful playback of the script, and the Citrix administrator can change the setting on the server by using the Citrix server console illustrated below.

¹Citrix Agent for HP LoadRunner can be installed from the “Additional Features” DVD provided with LoadRunner 9.0.

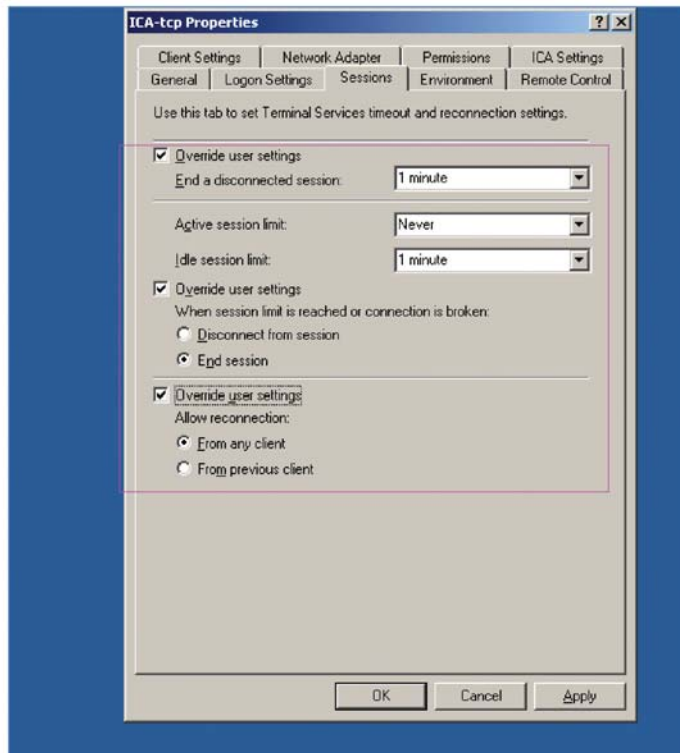


Figure 1 Citrix server console

ICA files

There two ways that HP LoadRunner establishes a connection with the Citrix server during recording and playback. The recording options of VUGen provide an option to enter the server/user details or use the Citrix ICA to connect to the Citrix server. The ICA file contains configuration information for the application accessed through the Citrix client; Citrix Administrators can provide the ICA file.

The ICA file should conform to the following format:

```
[WFClient]
Version=2
TcpBrowserAddress=10.2.248.109
PersistentCachePath=C:\Documents and Settings\sunilk\Application Data\ICAClient\Cache

[ApplicationServers]
Team Status=

[Team Status]
Address=Team Status
InitialProgram=#Team Status
ClientAudio=On
AudioBandwidthLimit=2
Compress=On
TWIMode=On
DesiredHRES=800
DesiredVRES=600
DesiredColor=4
TransportDriver=TCP/IP
WinStationDriver=ICA 3.0
ScreenPercent=0
Username=administrator
Password=00054c3b122d15
Domain=corpdev1
```

Synchronization

What is synchronization?

During the script execution it is important to synchronize the user steps with the application response to ensure successful playback of the script. Synchronization allows the virtual user to wait for a certain event to occur before moving onto the next step. This is especially useful when clicking on objects inside the window. Synchronization in the script will make sure that a window or object is returned by the Citrix server, and available for the user to execute the next step. For example, it is necessary to ensure that the “Create Customer” form window appears on the screen before you can enter customer information.

Adding synchronization during recording

Synchronization can be added into the script during recording. It can be inserted for text on the screen or the bitmap. The recording toolbar provides two buttons at the end. The first button allows the synchronization function to be interested on the bitmap, and the second button for the text. Users can click on any of these buttons depending on the type of synchronization needed, and then highlight the bitmap area or text on the screen during the recording. This will insert a step (Tree view) or function (Script view) within the script that will execute during the playback to ensure that the selected object appears on the screen before the virtual user can move to the next step in the script.

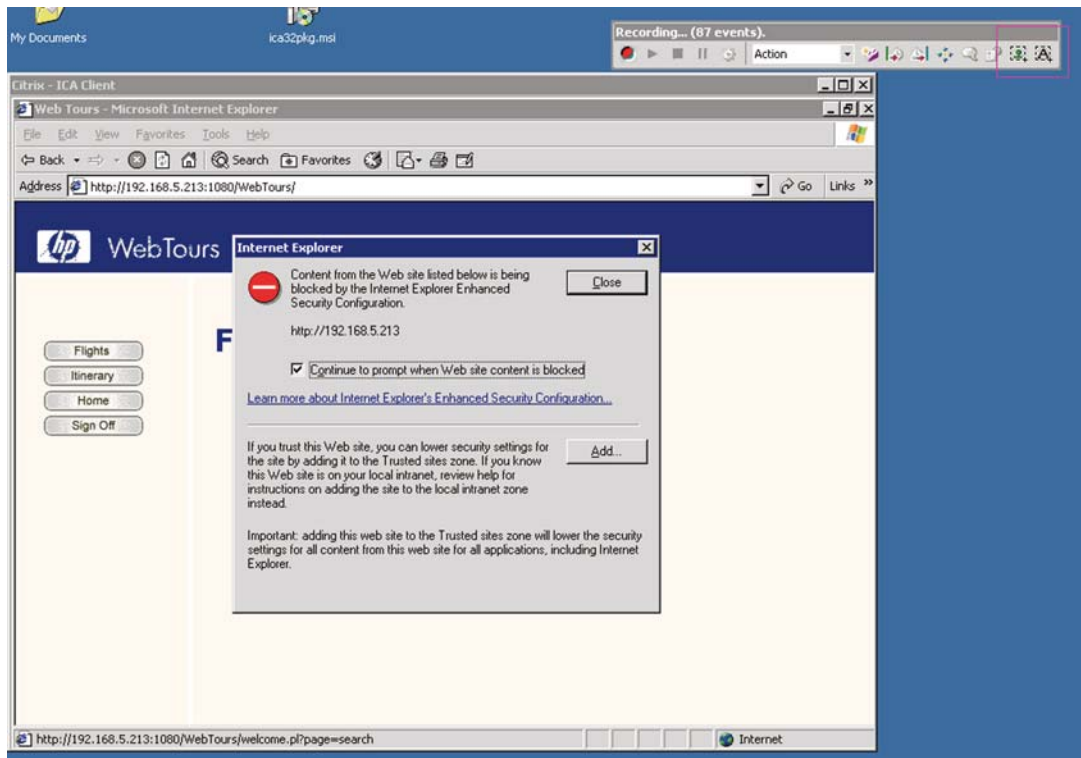


Figure 2 Shows the VUGen recording toolbar with the synchronization buttons

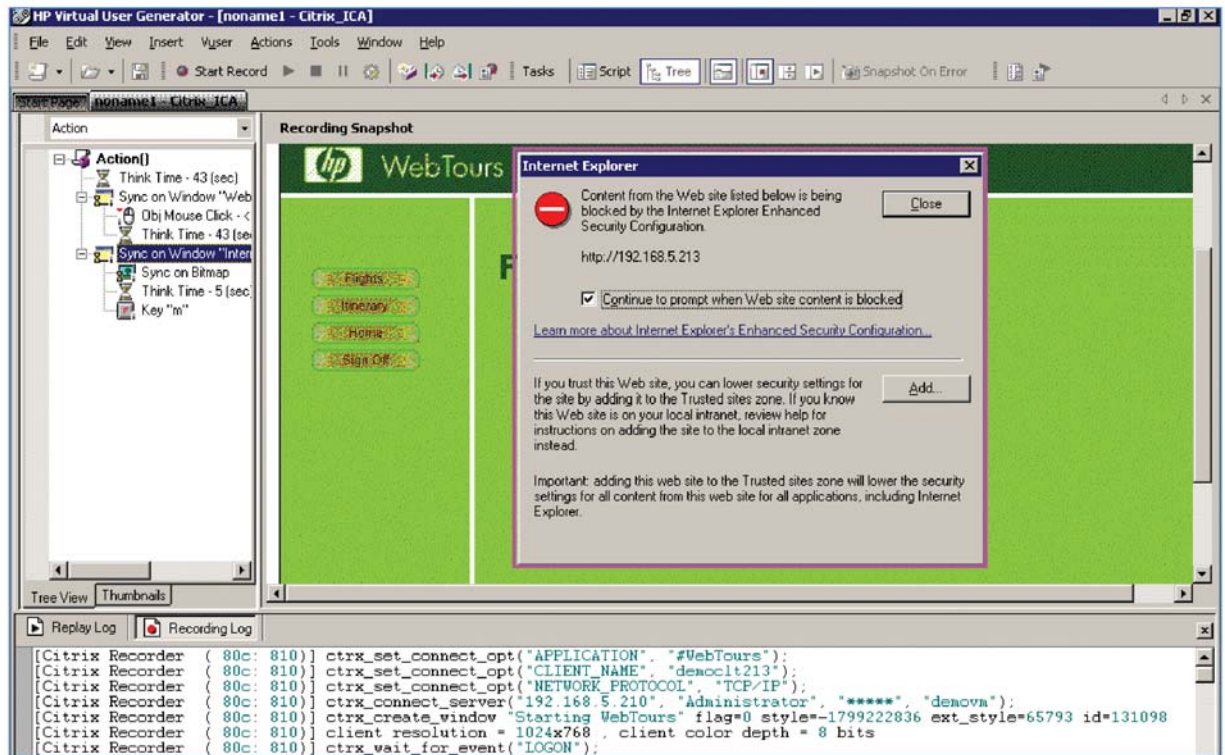


Figure 3 Shows the synchronization step on the left with the bitmap snapshot in VUGen (Tree view)

Adding synchronization after recording

Synchronization can be added after the recording by going into the Tree view of the script and clicking on the step with the object snapshot that needs to be synchronized. Once the snapshot is displayed, right-click on the object and select the type of synchronization that needs to be inserted. There are two different options of synchronization available in HP LoadRunner 9.0 software after the recording:

- Insert Sync on Text – waits until the specified text is displayed on the screen
- Insert Sync on Object Info – waits until an objects attributes have the specified values

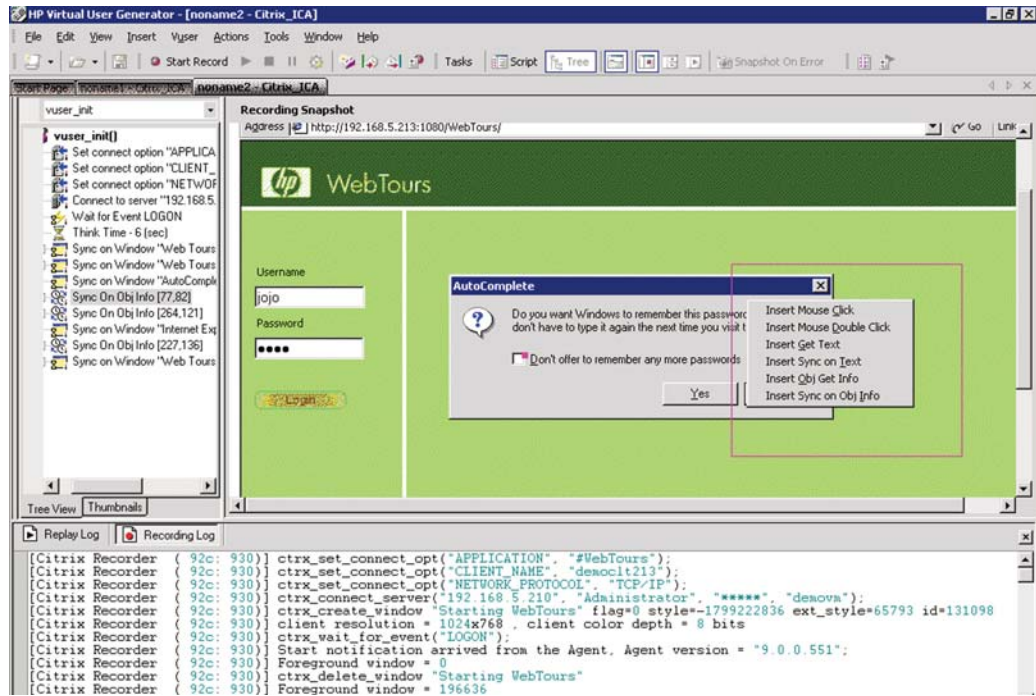


Figure 4 Right-clicking on the object within the snapshot window will display the list of sync options

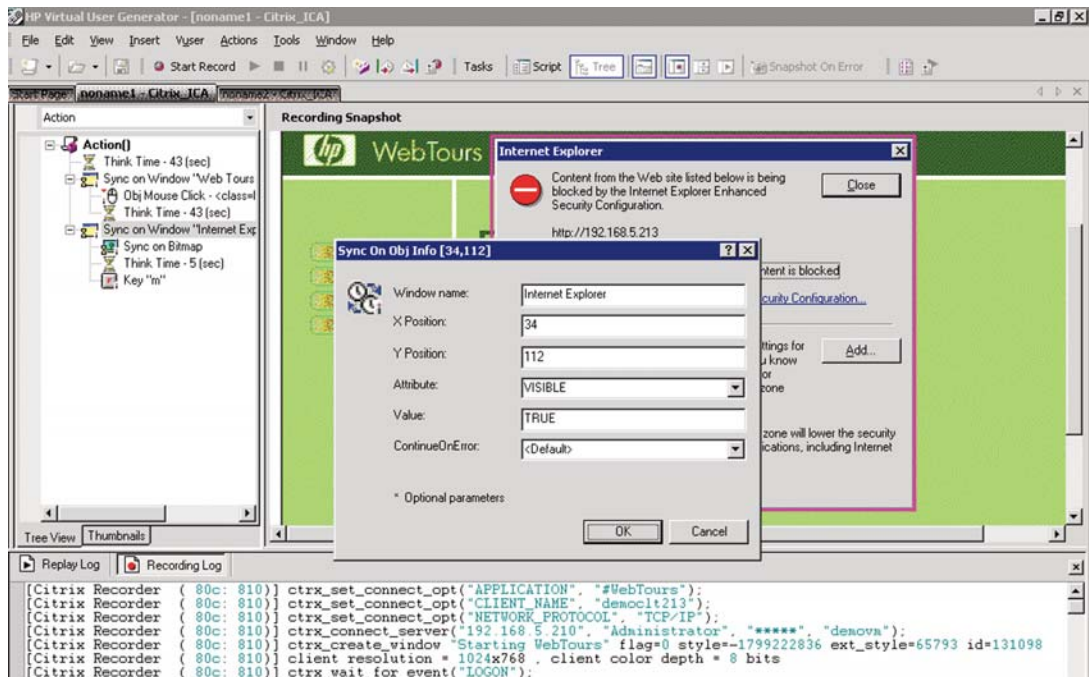
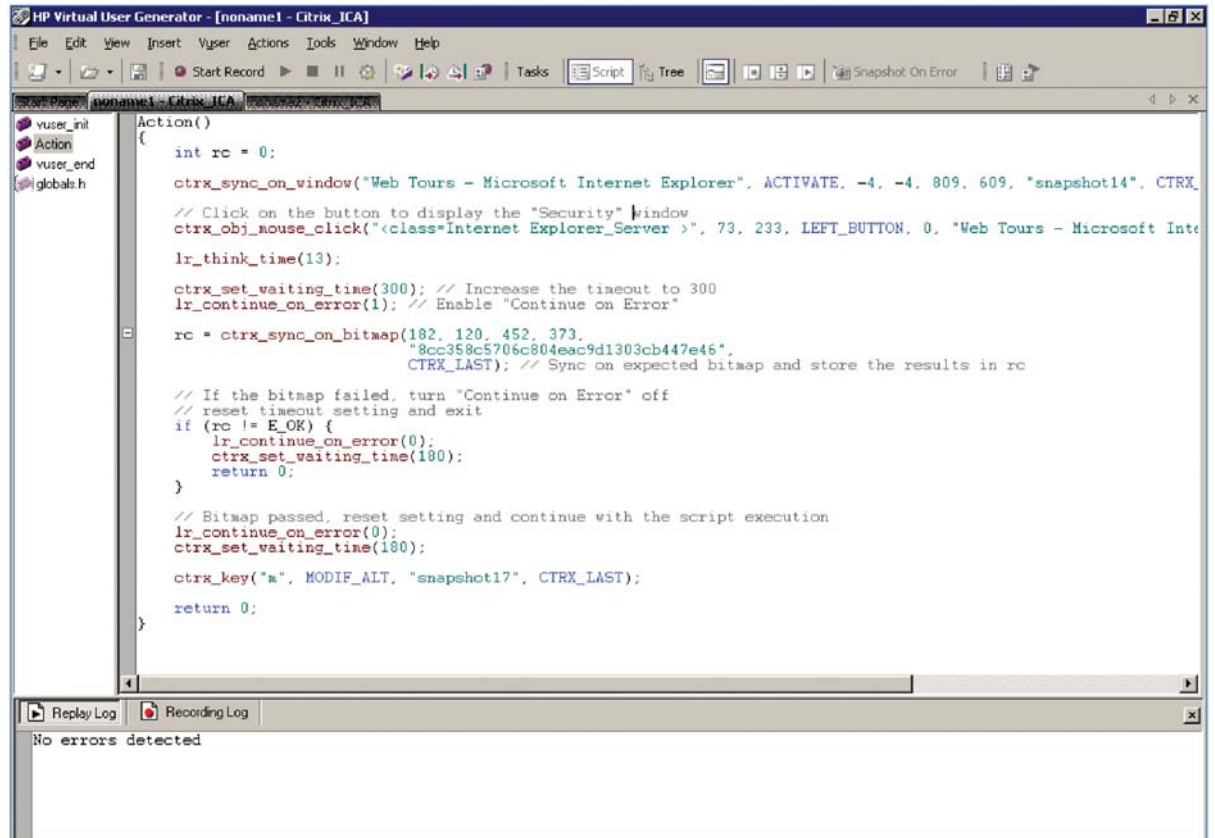


Figure 5 Displays the properties window for Insert Sync on Object

Adding logic to the synchronization

HP LoadRunner allows the user to add logic to the script programmatically by storing the return code of the synchronization function into an integer variable declared within the script. The return code values are standard Citrix values that can be found under the help section (i.e. "E_OK"). The example below provides a snapshot of the code. The virtual user clicks on the button and waits for the bitmap synchronization. If the expected value returns from the server, the script will continue to the next step. Otherwise, it will reset timeout and continue on error value and move onto the next action. HP LoadRunner provides a powerful flexibility to make logical decisions via coding within the script, and allows for the appropriate actions to be taken accordingly.



```
HP Virtual User Generator - [noname1 - Citrix_ICA]
File Edit View Insert Vuser Actions Tools Window Help
Start Record [Icons] Tasks [Script] [Tree] [Snapshot On Error]
noname1 - Citrix_ICA
vuser_init
Action
vuser_end
globals.h

Action()
{
    int rc = 0;

    ctrx_sync_on_window("Web Tours - Microsoft Internet Explorer", ACTIVATE, -4, -4, 809, 609, "snapshot14", CTRX_
    // Click on the button to display the "Security" window
    ctrx_obj_mouse_click("<class=Internet Explorer_Server >", 73, 233, LEFT_BUTTON, 0, "Web Tours - Microsoft Inte
    lr_think_time(13);

    ctrx_set_waiting_time(300); // Increase the timeout to 300
    lr_continue_on_error(1); // Enable "Continue on Error"

    rc = ctrx_sync_on_bitmap(182, 120, 452, 373,
        "8cc358c5706c804eac9d1303cb447e46",
        CTRX_LAST); // Sync on expected bitmap and store the results in rc

    // If the bitmap failed, turn "Continue on Error" off
    // reset timeout setting and exit
    if (rc != E_OK) {
        lr_continue_on_error(0);
        ctrx_set_waiting_time(180);
        return 0;
    }

    // Bitmap passed, reset setting and continue with the script execution
    lr_continue_on_error(0);
    ctrx_set_waiting_time(180);

    ctrx_key("m", MODIF_ALT, "snapshot17", CTRX_LAST);

    return 0;
}

Replay Log Recording Log
No errors detected
```

Figure 6 Citrix script with logical code added to perform certain functions depending on the return value

Wildcard

Wildcards, or asterisks (*), can be used in defining window names. This is very useful when the window name is dynamic and changes during the reply. Wildcards can be inserted at any location within the text string as a prefix and/or suffix.

```
ACTION()
{
    int rc = 0;

    ctrx_sync_on_window("* - Microsoft Internet Explorer", ACTIVATE, -4, -4, 809, 609, "snapshot14", CTRX_LAST);
}
```

Figure 7 Wildcards can be inserted as a prefix or suffix within the text string

Best practices for Citrix script development

Recording options

It is recommended that the recording options are set to default as these options remain the same for most Citrix implementations. However, there are a few recording options listed below that should be taken into consideration before recording the script.

- Windows sizing under the Configuration tab – the window sizing configuration should be identical to the display setting that is used for the script recording machine (VUGen) and the load generator. This setting enables HP LoadRunner to display the windows in the size selected from the dropdown menu.

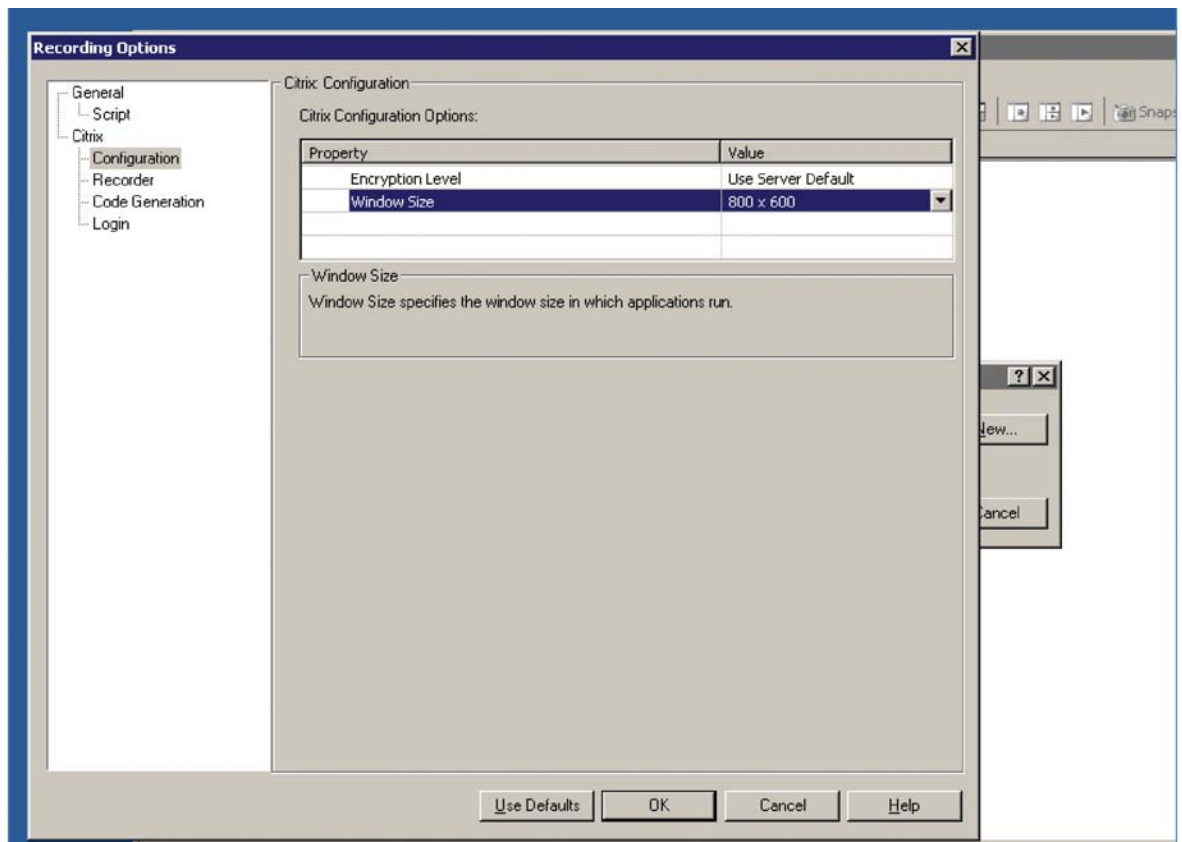


Figure 8 Recording Options -> Configuration tab

- Window name under Recorder tab – this option allows the script to be recorded with a pre-determined prefix or suffix for the window name. This is especially useful when the window name returned by the Citrix server is dynamic in nature. By using this option, an asterisk (*) can be inserted automatically by VUGen within each function that uses the window name. This can be achieved manually when the need to add a prefix or suffix is minimal.
- Save snapshot under Recorder tab – saving snapshot during recording is a good practice and should be utilized whenever possible. This setting slows the recording process and allows the user to correlate the user step within the script with the visual snapshot.

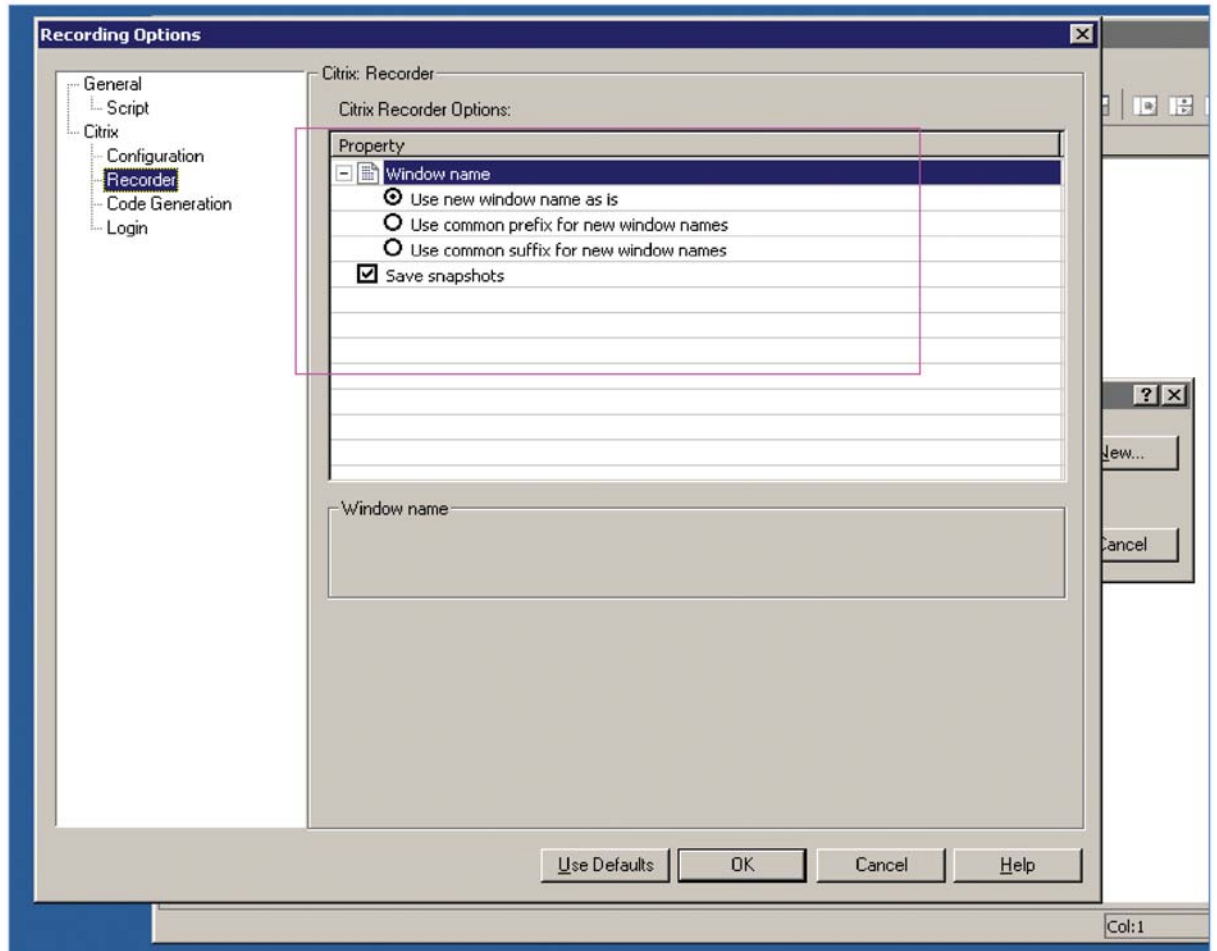


Figure 9 Recording Options -> Recorder tab

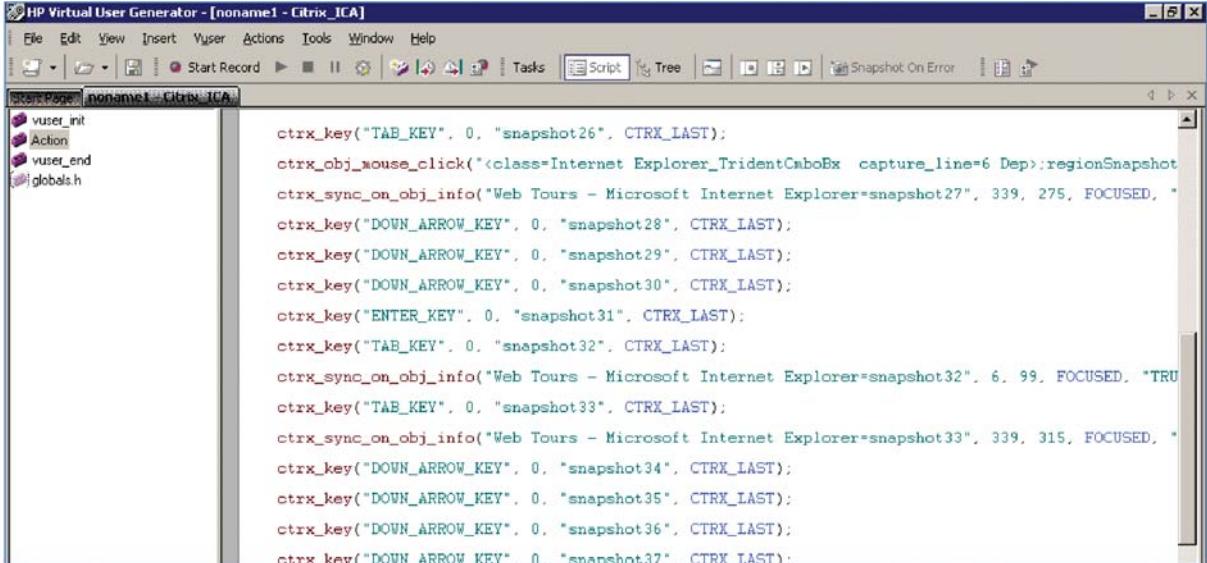
Using the keyboard versus a mouse

During script recording, HP LoadRunner captures mouse clicks and key strokes as they are emulated at the time of playback to simulate users performing an action. The mouse clicks are recorded based on the object name and the X,Y coordinates of the screen and accordingly, during the play back, HP LoadRunner sends instructions to the Citrix server to emulate these mouse clicks based on the object name and the X,Y coordinate. The emulation of the virtual user based on the X,Y coordinates can be unreliable and if the coordinates do not match accurately, the script will fail.

The line of code provided in the example below shows a script trying to click on a Close button with 331 as an X coordinate and 308 as a Y coordinate. If the parent window of the object containing the Close button appears at a different location on the screen during the playback of the script, it will change the X,Y coordinate of the Close button object itself. As a result of this change, the test will fail.

```
ctrx_obj_mouse_click("<class=Button text=Close>", 331, 308,LEFT_BUTTON,0,CTRX_LAST);???
```

The solution is to use the keyboard instead of the mouse whenever possible during the recording. This avoids the recording of X,Y screen coordinates in the script and sends the key strokes directly to the Citrix server. The code in Figure 4 provides an example of how the key strokes are recorded in the script.



The screenshot shows the HP Virtual User Generator (VUGen) interface. The main window displays a script with the following code:

```
ctrx_key("TAB_KEY", 0, "snapshot26", CTRX_LAST);
ctrx_obj_mouse_click("<class=Internet Explorer_TridentCaboEx capture_line=6 Dep>:regionSnapshot
ctrx_sync_on_obj_info("Web Tours - Microsoft Internet Explorer=snapshot27", 339, 275, FOCUSED, "
ctrx_key("DOWN_ARROW_KEY", 0, "snapshot28", CTRX_LAST);
ctrx_key("DOWN_ARROW_KEY", 0, "snapshot29", CTRX_LAST);
ctrx_key("DOWN_ARROW_KEY", 0, "snapshot30", CTRX_LAST);
ctrx_key("ENTER_KEY", 0, "snapshot31", CTRX_LAST);
ctrx_key("TAB_KEY", 0, "snapshot32", CTRX_LAST);
ctrx_sync_on_obj_info("Web Tours - Microsoft Internet Explorer=snapshot32", 6, 99, FOCUSED, "TRU
ctrx_key("TAB_KEY", 0, "snapshot33", CTRX_LAST);
ctrx_sync_on_obj_info("Web Tours - Microsoft Internet Explorer=snapshot33", 339, 315, FOCUSED, "
ctrx_key("DOWN_ARROW_KEY", 0, "snapshot34", CTRX_LAST);
ctrx_key("DOWN_ARROW_KEY", 0, "snapshot35", CTRX_LAST);
ctrx_key("DOWN_ARROW_KEY", 0, "snapshot36", CTRX_LAST);
ctrx_key("DOWN_ARROW_KEY", 0, "snapshot37", CTRX_LAST);
```

Figure 10 VUGen displaying a Citrix script recorded using the keyboard instead of mouse clicks when possible

Window resizing

It is recommended to not resize the window during recording although it is supported by HP LoadRunner. The resizing changes the screen coordinates of the objects that are contained in the window. The window can be resized after the recording by going into Tree view, selecting the appropriate step, right-clicking and selecting Properties. The Properties window will be displayed with the width and height coordinates that can be changed as needed.

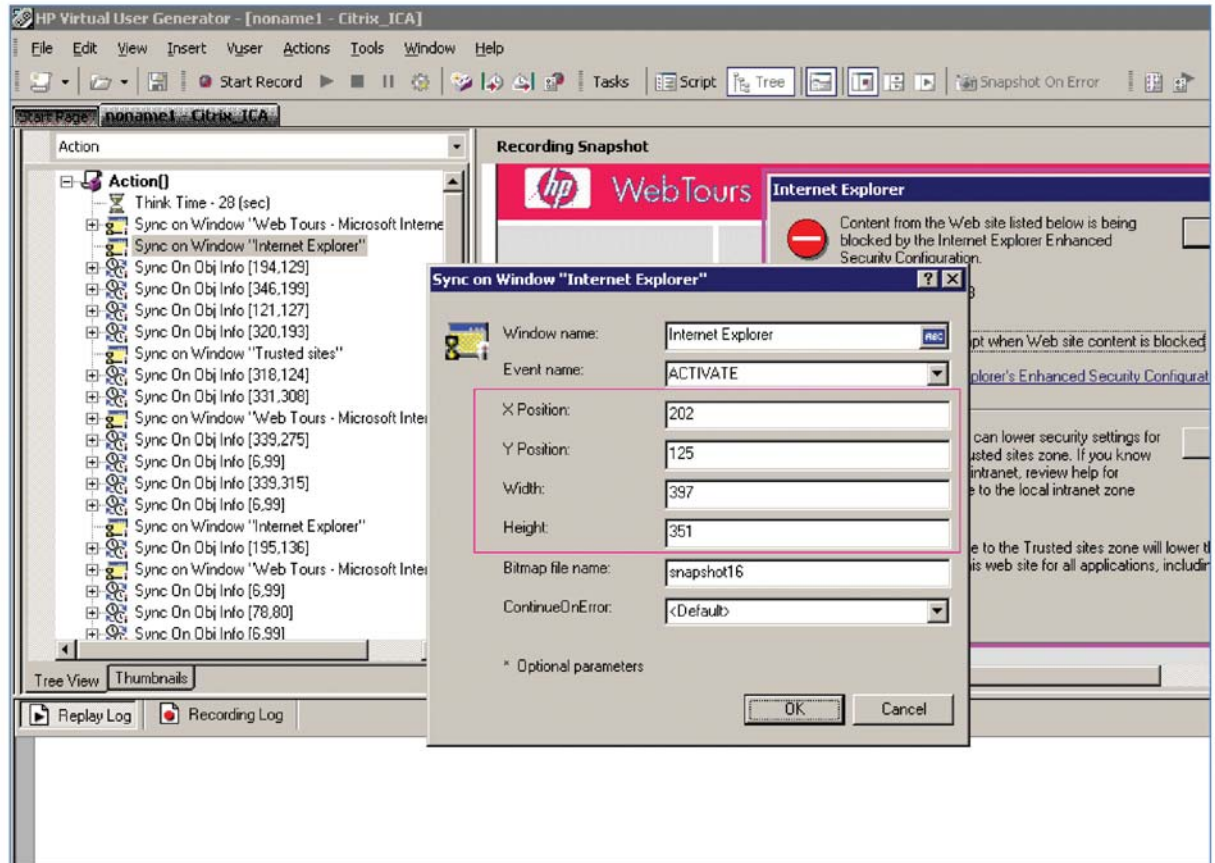


Figure 11 VUGen displaying a Citrix script in Tree view with the Properties window of the Sync on Window step

Display settings and color resolution

The display setting of the load generators and the screen resolution should be exactly the same as when the script was recorded. This is a common problem that occurs during the playback of Citrix scripts. The recording resolution and playback resolution should always be identical for the successful playback of the script.

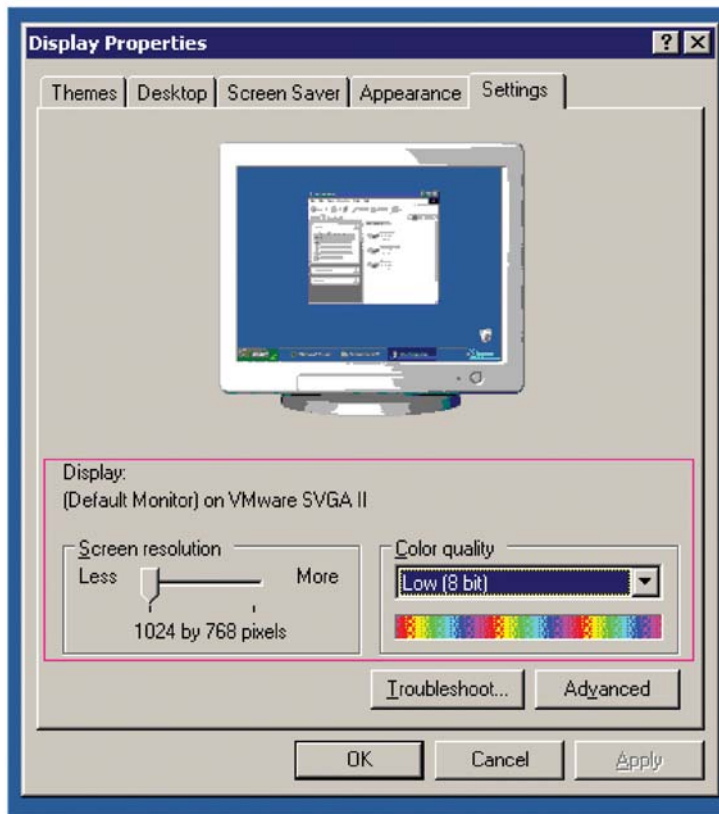


Figure 12 Display settings

Recording a complete session (running a clean session)

When recording a script, always ensure to perform the complete business process from beginning to end. The end should include the session clean up by logging out of the application. This will make sure that the session is cleaned up upon exit, and all windows are closed. The start and end conditions should always be the same so that the script can start where it ended.

Using various Action sections

Action sections of scripts should be utilized to separate the login from main business process, and end with logout. Login should be recorded in the `user_init ()` action, while the main business process or transaction is recorded in one or multiple `Action ()` sections. The logout should always be recorded in the `user_end ()` section of this script. It is a good practice to separate the main business process in small segments of multiple action sections for the ease of maintaining and managing the script. Action sections should be renamed to the logical name that corresponds to the business process for intuitiveness.

Timeout settings

Global timeout settings can be adjusted through the script Run-time Settings and Timing tab. Connect time allows the user to wait idly after establishing the connection for the period specified. This setting should be left to default unless the Citrix authentication is slow in response.

The waiting time setting applies globally to all synchronizations within the script and can be changed programmatically by using the `ctx_set_waiting_time()` function. Depending on the server response, the wait time should be adjusted on an "as needed" basis.

The typing rate setting defines the delay for the virtual user between the key strokes, and is especially useful for slowing the user interaction with the client or when working with limited bandwidth.

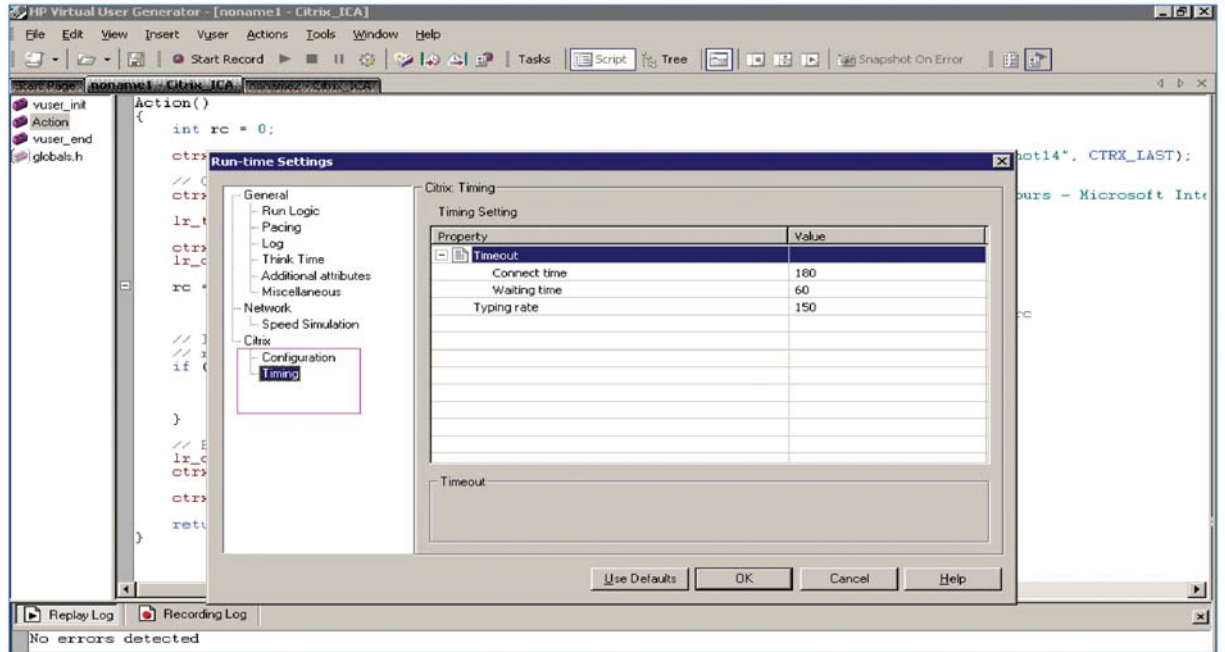


Figure 13 Run-time Settings to adjust the timeout setting window

Continue on Error setting

The Continue on Error option allows the script or individual functions within the script to continue when the error occurs or the synchronization fails. Continue on Error can be used for individual functions either by the Tree view or by adding the `lr_continue_on_error()` function prior to the synchronization function, allowing the virtual user to continue when the specific steps fail while further action can be taken such as trapping the error or sending a message to the log file for debugging purposes.

```

Action() [Start recording your application (Ctrl+R)]
{
    int rc = 0;

    ctrx_sync_on_window("* - Microsoft Internet Explorer", ACTIVATE, -4, -4, 809, 609, "snapshot14", CTRX_LAS
    // Click on the button to display the "Security" window
    ctrx_obj_mouse_click("<class=Internet Explorer_Server >", 73, 233, LEFT_BUTTON, 0, "Web Tours - Microsoft
    lr_think_time(13);

    ctrx_set_waiting_time(300); // Increase the timeout to 300
    lr_continue_on_error(1); // Enable "Continue on Error"

    rc = ctrx_sync_on_bitmap(182, 120, 452, 373,
                           "8cc358c5706c804eac9d1303cb447e46",
                           CTRX_LAST); // Sync on expected bitmap and store the results in rc

    // If the bitmap failed, turn "Continue on Error" off
    // reset timeout setting and exit
    if (rc != E_OK) {
        lr_continue_on_error(0);
        ctrx_set_waiting_time(180);
        return 0;
    }

    // Bitmap passed, reset setting and continue with the script execution
    lr_continue_on_error(0);
    ctrx_set_waiting_time(180);

    ctrx_key("n", MODIF_ALT, "snapshot17", CTRX_LAST);

    return 0;
}

```

Figure 14 Shows Continue on Error manually inserted within the script for error handling

Replay tips and tricks

Setting Think Time

Citrix virtual users work similarly to real users and as a result, the wait time between various steps is very important. This allows the virtual user to wait for the server response, and then move onto the next step. Thus, the Think Time plays a critical role in stabilizing the communication between the client and server. Think Time should always be enabled in order for the user to successfully complete the playback. It allows the virtual user to receive the response from the server, and complete the screen rendering before moving on the next step. The Think Time in the script should always be comparable to the real user. HP LoadRunner provides a Run-time setting to randomize the Think Time, which applies to the virtual users and affects all Think Time within the script. Depending on the goals of the load test and the real world condition of how the user interacts with the system, this setting should be typically set to 100–150 percent. Ignoring Think Time can have an adverse affect on the script and the application, and more then likely the virtual users will fail under load.

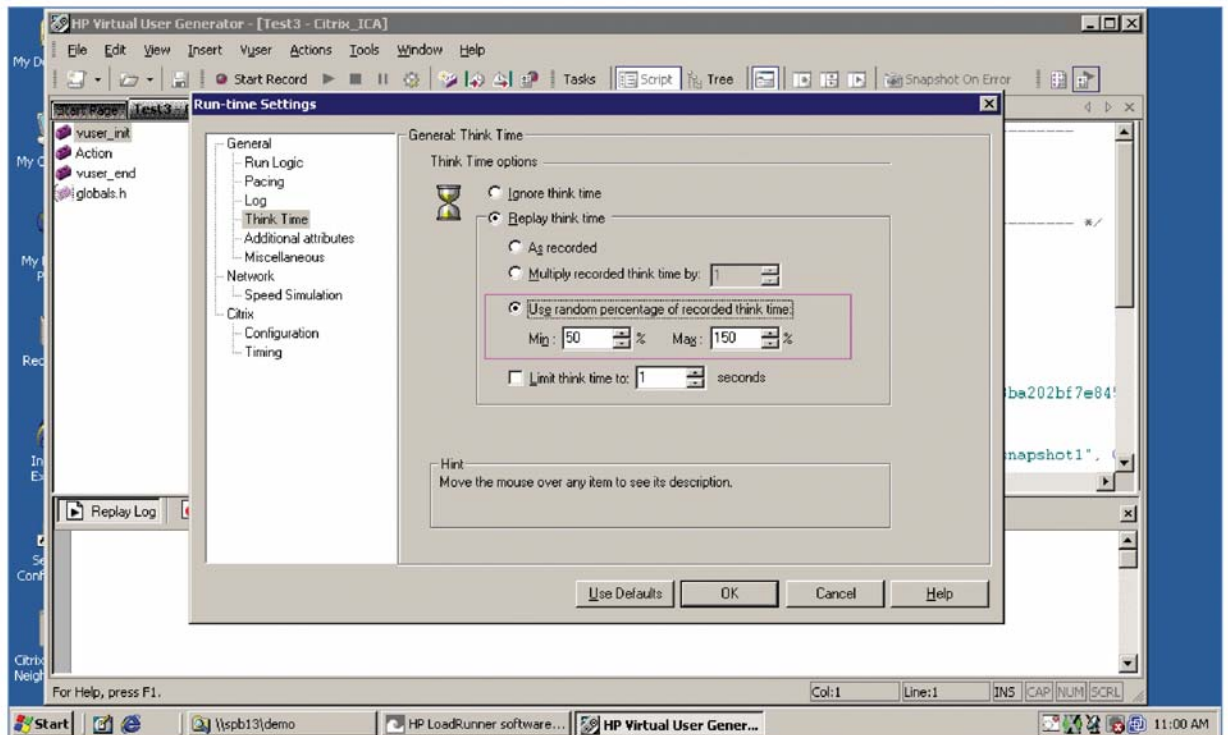


Figure 15 Shows Think Time randomization option from the Run-time Settings tab

Generate Snapshot on Error

Snapshot on Error is a very useful tool that helps in debugging the scripts or identifying a point of failure when the virtual user fails during the execution. The Generate Snapshot on Error option for the script can be enabled through the Run-time Settings and Miscellaneous tab. During the execution, if the user fails, HP LoadRunner will capture the screen snapshot and store it in the data folder of the script or the virtual user if running a scenario. The snapshot can be accessed by going into Output Log, and by clicking on the Camera button at the point of failure. It is a good practice to enable Generate Snapshot on Error for a few users in the scenario to avoid overhead on the load generator. It is also a very useful tool for debugging the script in VUGen during script development.

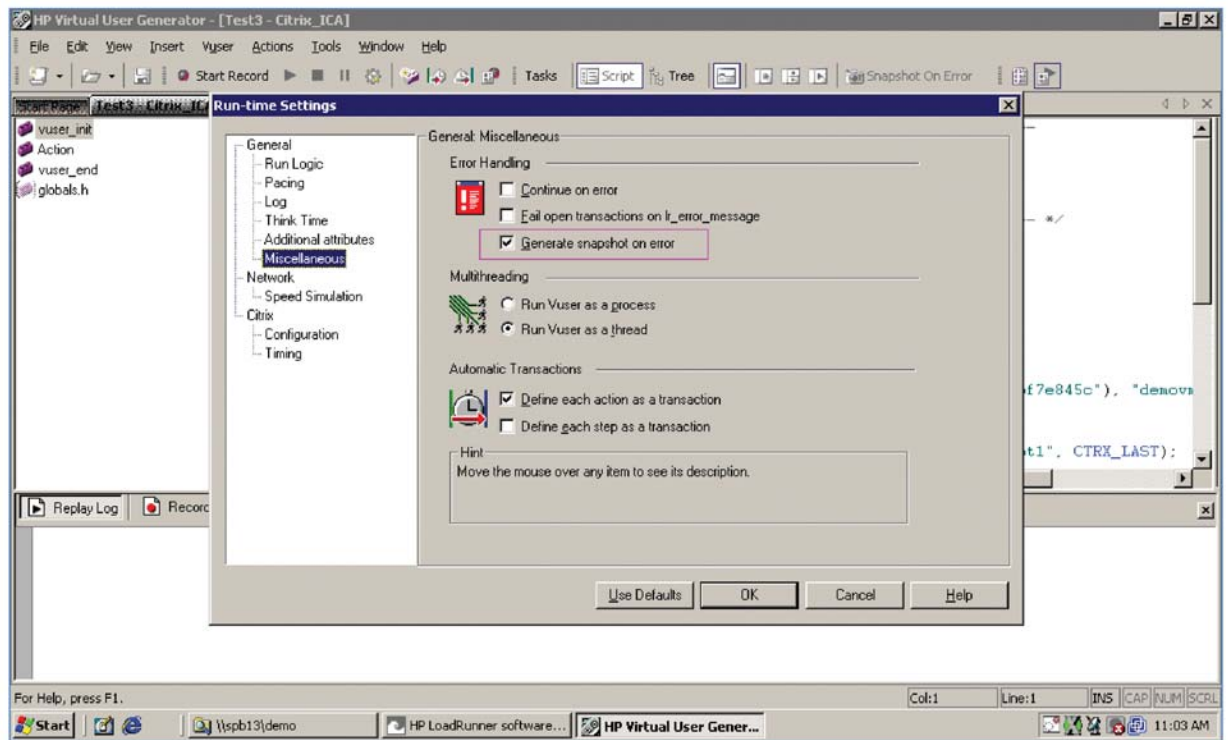


Figure 16 Shows Generate Snapshot on Error option of the Run-time Settings for error handling

GDI resources

Citrix virtual users use graphics device interface (GDI) resources on the load generators. The GDI resources are limited on each machine, and are determined by the operating system that is being used. GDI resources are also tied to the individual session running on the system. The Microsoft Windows 2000 operating system has a maximum of 16,384 GDI resources available. However, these resources are also used by other applications running on the machine, so the numbers of users that can be executed are primarily determined by the available GDI resource. It is recommended to limit the number of maximum users to 40 per Load Generator provided there is enough memory and CPU available.

In order to increase the number of users on the load generator, you can open a remote desktop protocol (RDP) connection and connect to the individual RDP sessions as Load Generator. This can be achieved by checking "Enabling Terminal Services" in Load Generator Information from the HP LoadRunner Controller. Each RDP session maintains its own GDI resources; if three RDP sessions are established on the Load Generator, each can execute 40 users. Starting an RDP session on the Load Generator will consume memory and CPU on that machine, thus limiting the number of users in other areas.

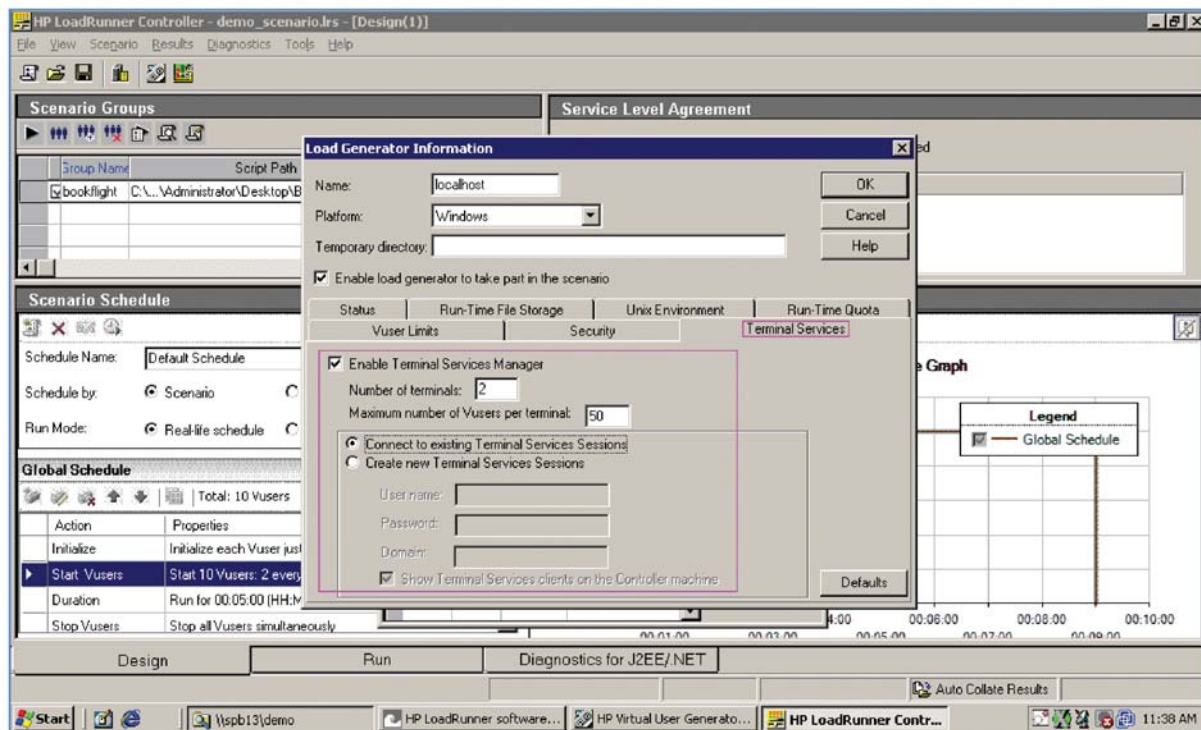


Figure 16 Shows Generate Snapshot on Error option of the Run-time Settings for error handling

Debugging tips and tricks

Logging

Logging is an important tool for debugging the script or looking at the error message during scenario execution. Logging does, however, generate overhead on the Load Generator—especially during a load with many users. The best practice for using the log setting for load testing of other application applies to Citrix as well. It is recommended to create a separate group in the Controller for 1–5 users, and turn the logging on for those users only. This avoids the overhead on the load generator as small subsets of the users are logging. Logging also allows you to look at the Snapshot on Error screen in case of an error.

Show client during replay from Controller

During the scenario execution, virtual users can be played back in the Run-time Viewer to display user activity. This functionality is very useful for debugging purposes and can be used to examine the user behavior. Displaying the virtual user replay during scenario execution will impact the scalability of the test and affect the load generator due to the overhead it generates, hence only one user per Load Generator should be replayed back to observe the behavior.

To enable the virtual user replay during scenario execution, open the Vuser Details dialog box, click More and type `-lr_citrix_vuser_view` in the command line box. You can provide the virtual user ID number to display specific users by adding `<VUSER ID #>` at the end.

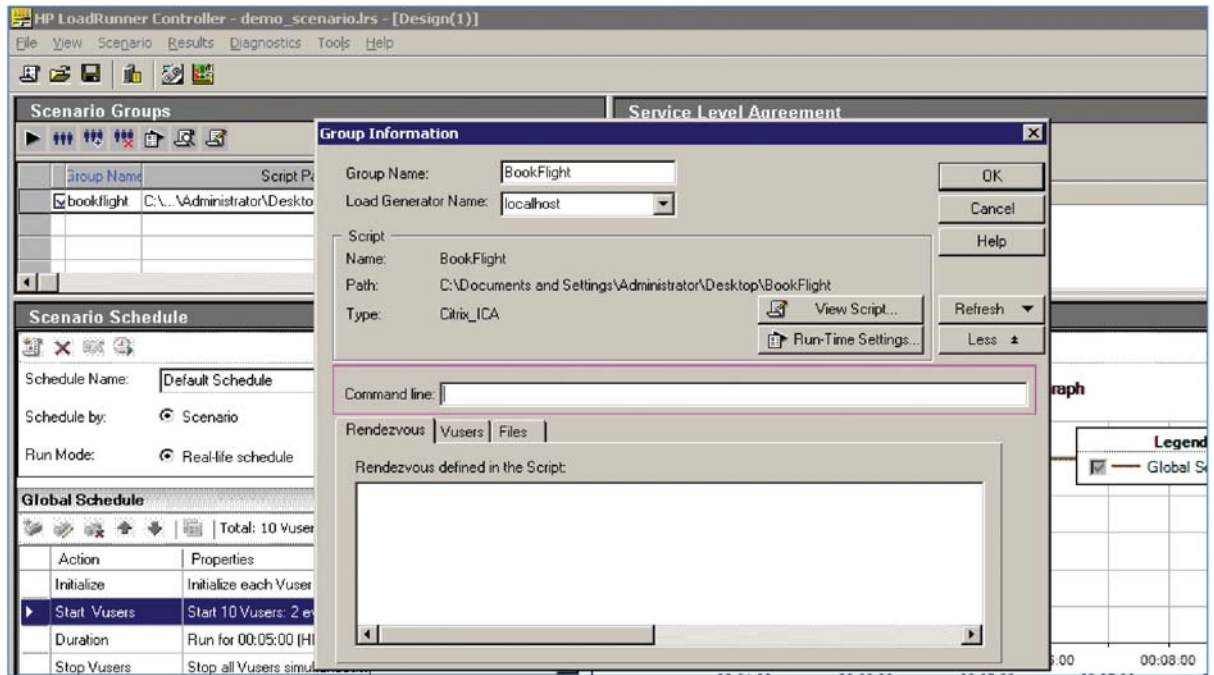


Figure 18 Shows Command line option to pass arguments to the virtual users from the Details option of the Vuser group

Best practices for scenario execution of Citrix Presentation Server 4.5 virtual users with HP LoadRunner 9.0 software

Load Generator considerations

Display settings

As noted above, it is necessary to be vigilant about the display settings on the load generators. The Load Generator display setting should match the display setting when the script was recorded, otherwise the script will fail.

Location

Application delivery using Citrix is increasingly global in scope. One of the primary goals of a Citrix testing project is to assess the contribution to network bandwidth of typical Citrix user sessions. Therefore, it is not unusual to install load generators in multiple remote locations. This can present logistical challenges that impinge on the technical readiness of a test. A single test can be “corrupted” by failure of a remote laptop that has been deployed as a load generator but underwent a reboot or perhaps an inadvertent shutdown in the middle of night.

Therefore, prior to each significant test, it is important to have a checkout run with a limited number of scripts and/or users in order to re-validate the readiness of the remote agents. There should also be remote desktop access in order to correct any issues that might be revealed.

Concurrency

The Citrix environment is remarkably resilient to network disruptions. Each connection is checked against existing sessions that may have been orphaned by a prior disruption and the user is re-connected to that pre-existing session. For load tests, this has two impacts:

- When a test fails, for whatever reason, it is not unusual to have perhaps many user sessions disconnected without benefit of a graceful disconnect. Therefore, it is best to set the Citrix server setting that governs the timeout for disconnected sessions to the minimum (one minute). If this is not possible, you may need to work with the Citrix administrator to remove any dangling sessions.
- Secondly, for the same reason, it is necessary to have unique Citrix user data so that each simulated client is detected as a distinct user. Otherwise, connection failures and a variety of unexpected disconnects may occur.

Ramp rates

It is important to establish the maximum rate at which Citrix users may be added. The Citrix server will have a small number of “idle” workstations available for use by new connections. However, it is not unusual during a test’s initialization to overwhelm the Citrix infrastructure with requests for new workstations—especially when a high load is added to an unreasonable ramp rate.

Therefore, initial “shakedown” testing should be done in order to establish the maximum rate at which users can be added without undue stress. If you are using the ramp period to establish the “performance under load” graphs, be sure to permit connected users to begin the action iterations. Getting good connections and keeping those connections intact under a heavy load can be a significant hurdle to overcome. Keeping the ramp rate in a “safe” zone is a must.

This may take a significant toll on the time required to run a test. Simulating 1000 users may take hours to start up.

Logging and viewable Citrix clients

Debugging

As with most testing, the appropriate use of logging is an important consideration, especially early in the shakedown process. Frequently, the application may behave differently (extra pop-ups, validation failures, etc.) and it can be a challenge to determine the points of failure—especially when the application data has not been completely vetted. Enabling parameterization logging can at least help pinpoint the impact of data.

By turning up the logging during the shakedown runs, you can discern a number of issues. For instance, during connection the display settings on the client are logged and can be the first sign that a load generator has been misconfigured.

In general, Citrix logging is less verbose than other HP LoadRunner protocols, but should still be minimized to avoid excessive overhead on the network, as well as the load generators themselves. It is recommended to create a separate group in the Controller for 1–5 users and turn the logging on for those users only. This avoids the overhead on the load generator as a small subset of the users is logging. Logging also allows you to look at the Snapshot on Error screen in case an error is provided.

Viewing Citrix clients

When scripts tend to fail more frequently (or always) on specific load generators, it sometimes means not being able to visually follow the execution of the test. The `-lr_citrix_vuser_view` command line parameter can be added to the Groups detail page in the Controller. This setting instructs the Agent to render the execution of the Citrix client session on the desktop. By going to that desktop, you can see just what is happening on the screen. It is best to restrict the number of running sessions to just a few Vusers since the desktop itself can become difficult to manage.

To learn more, visit www.hp.com
www.citrix.com or www.genilogix.com

© 2007 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Microsoft is a U.S. registered trademark of Microsoft Corporation. Windows is a U.S. registered trademark of Microsoft Corporation.

4AA1-5197ENW, August 2007

