Monitoring NetScaler Web AppFlow

eG Innovations Product Documentation

www.eginnovations.com

Total Performance Visibility

Table of Contents

Table of Figures

Chapter 1: Introduction to NetScaler Web AppFlow Monitoring

The NetScaler appliance is a central point of control for all application traffic in the data center. It collects flow and user-session level information valuable for application performance monitoring, analytics, and business intelligence applications. It also collects web page performance data and database information.

AppFlow is a new flow export standard for finding data pertaining to applications and transactions in the network infrastructure.

AppFlow records contain standard NetFlow or IPFIX information, such as time stamps for the beginning and end of a flow, packet count, and byte count. AppFlow records also contain application-level information (such as HTTP URLs, HTTP request methods and response-status codes, server response time, and latency).

You can configure AppFlow on a NetScaler device. This ensures that AppFlow records are transmitted to an IPv4 collector, using the Internet Protocol Flow Information eXport (IPFIX) format. The collector is where these real-time analytics can be aggregated for providing administrators with greater visibility into application traffic utilization and performance.

eG Enterprise supports AppFlow monitoring. You can configure eG Enterprise to serve as an AppFlow collector for an AppFlow-enabled NetScaler device in your environment. This way, you can have eG Enterprise receive , process, and publish in its console, the application and session-level statistics that AppFlow provides. With the help of these analytics, administrators can closely scrutinize application/desktop session traffic (via HDX virtual channels) and traffic to/from web applications. In the process, administrators can measure the bandwidth used and latency experienced by user accesses to XenApp servers, XenDesktop virtual desktops, and web applications, and determine the root-cause of poor user experience.

For this purpose, eG Enterprise provides two specialized monitoring models - the *NetScaler HDX AppFlow* model and the *NetScaler Web AppFlow* model.

This document elaborates on the *NetScaler Web AppFlow* model only. For details regarding the *NetScaler HDX AppFlow* model, refer to the *Monitoring NetScaler HDX AppFlow document* .

The *NetScaler Web AppFlow* model aggregates the AppFlow metrics on web traffic that flows through web applications, and provides useful insights into performance web applications. Using this document , you can understand how to configure eG Enterprise as an AppFlow collector, what metrics it collects and reports for web traffic, and how to interpret these analytics.

1.1 How does eG Enterprise Perform AppFlow Monitoring?

To intercept and process AppFlow records exported by NetScaler, eG Enterprise offers an eG AppFlow Collector. The collector runs as a Windows service. Once started, the collector service starts listening for AppFlow records on UDP port 9996 (by default). The NetScaler device in the target infrastructure should be configured to export AppFlow records to the collector. The collector receives these records, processes them, and stores the processed data in binary files.

To analyze and aggregate the data stored in the binary files, the eG remote agent monitoring the NetScaler device serves as the eG AppFlow Analyzer/Aggregator. This agent periodically reads these binary files, pulls statistics on appflow, processes/aggregates these statistics on the basis of HDX channels, users, applications, application/desktop sessions etc., and reports the aggregated data to the eG manager. The eG manager then stores this information in the eG database.

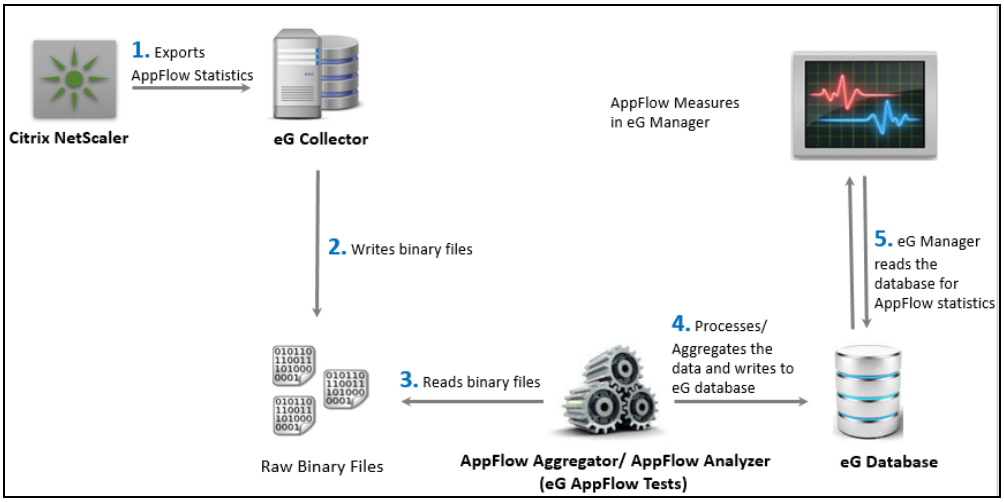


Figure 1.1: How the eG NetFlow Collector Works

For a NetScaler device, the eG manager also presents real-time metrics on traffic and bandwidth in the eG monitoring console using a specialized NetScaler HDX AppFlow monitoring model. If abnormalities are spotted during flow analysis, alerts are generated on this model.

1.2 Licensing

AppFlow monitoring by eG Enterprise is licensed by the number of eG external Agents used for collecting flow data from AppFlow-enabled devices. Licensing is NOT restricted by the number of devices/interfaces exporting flow data.

Each eG external Agent includes one AppFlow Collector, which supports the collection of up to 20,000 flows/second.

Chapter 2: Setting Up AppFlow Monitoring

The broad steps towards setting up AppFlow Monitoring are as follows:

1. Using the eG management console, manage the NetScaler device that will be generating AppFlow records, and assign a remote agent to it.
2. Deploy the eG AppFlow Collector on the same system that hosts the remote agent assigned to the NetScaler device at step 1 .
3. Open the UDP port (9996, by default) on the collector host, so that the collector can intercept and process AppFlow records; then, staeach of rt the collector.

This chapter discusses each of these steps in detail.

Section **2.1**

Section **2.2**

Section **2.3**

2.1 Deploying the eG AppFlow Collector

The next step is to deploy the eG AppFlow Collector. As mentioned already, the collector has to be created as a Windows service. Note that the **collector service has to be created on the same Windows host on which the remote agent monitoring the target NetScaler device has been deployed**.

To create the collector service on the remote agent host, follow the steps below:

1. Login to the Windows system hosting the eG remote agent.
2. Run the command prompt in elevated mode.
3. Go to the <EG_ INSTALL_ DIR>\Netflow\bin directory and execute the **CreateNetflowService.bat** batch file.
4. Successful execution of the batch file will result in the creation of a Windows service named **eGNetFlowAgent**. To confirm the creation of this service, open the **Services** window on the collector host. If you find the **eGNetflowAgent** service displayed therein (see Figure 2.1), it denotes that the collector has been successfully deployed.

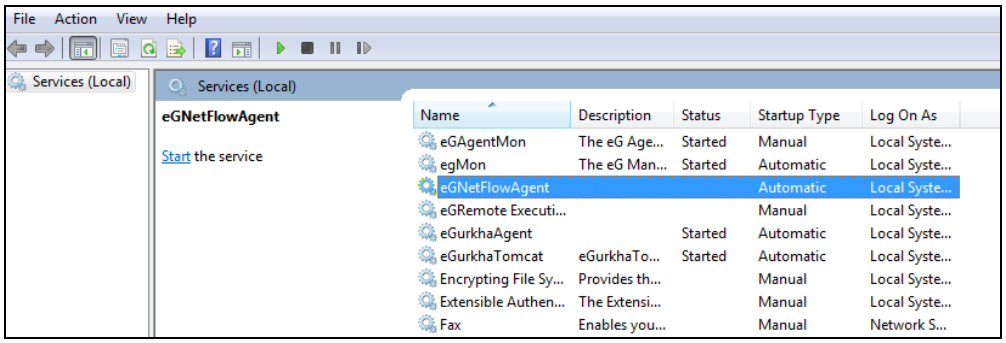


Figure 2.1: The eGNetflowAgent service displayed in the Services window

5. By default, the collector listens on UDP port 9996 for AppFlow records from the managed NetScaler device. To make sure that the AppFlow device communicates with the collector via this port, make sure you open this port on the firewall.
6. Once this is done, proceed to start the collector service. For this, right- click on the **eGNetFlowAgent** service in Figure 2.1 and select the **Start** option from the shortcut menu that appears.
7. On the other hand, if the default UDP port 9996 is already in use, then you will first have to change the listening port of the collector. In this case therefore, skip step 6 above; instead, follow the steps detailed in the Configuring the eG AppFlow Collector to Receive AppFlow Records topic.

2.2 Managing a Web AppFlow Device in eG Enterprise

To do this, follow the steps below:

1. Login to the eG administrative interface.
2. Follow the Infrastructure -> Components -> Add/Modify menu sequence. In the page that appears next, select *NetScaler Web AppFlow* as the **Component type** and click the **Add New Component** button to add a new component of that type. Figure 2.2 will then appear.

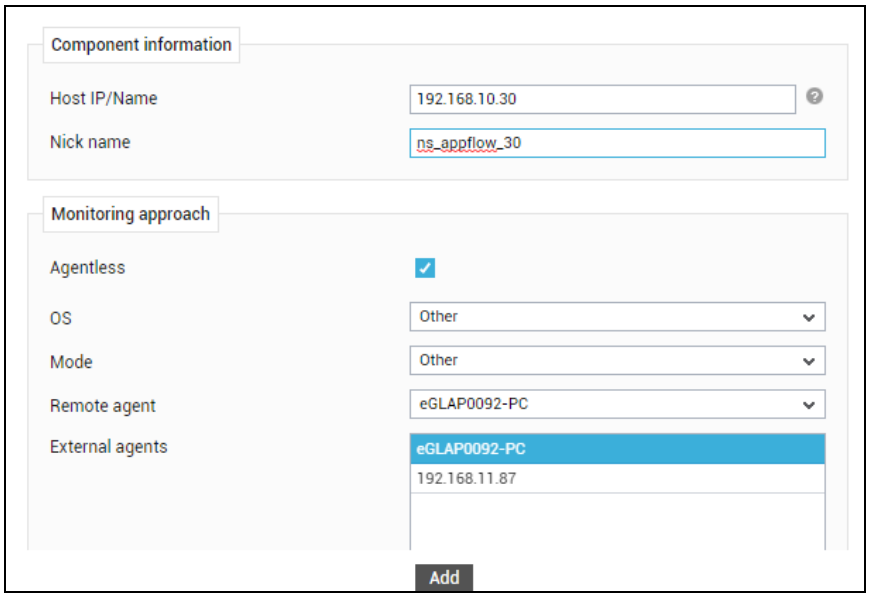


Figure 2.2: Adding a Web AppFlow Device

3. Specify the IP address and **Nick name** of the AppFlow-enabled NetScaler device to be monitored in Figure 2.2. Then, pick a **Remote Agent** from Figure 2.2 to assign to the AppFlow Device.

   Note:

   Select a remote agent that has been deployed on a Windows host for the purpose of AppFlow monitoring.

4. Finally, click the **Add** button.

2.3 Configuring the eG AppFlow Collector to Receive AppFlow Records

By default, the collector listens for AppFlow records on UDP port 9996. **Make sure you open that port on the firewall,** so that the managed AppFlow-enabled NetScaler device is able to communicate with the collector via that port. If the default port is already in use in your environment, then you can change the listening port of the collector. For this, do the following:

1. Login to the system hosting the eG AppFlow Collector.
2. Edit the **NetFlow.properties** file in the <EG_INSTALL_DIR>\Netflow\config directory.
3. Look for the **net.bind.port** entry in that file. This will be set to *9996* by default. Change the port number against that entry and save the file.

4. Finally, start the collector service. For that open the Services window, right-click on the **eGNetFlowAgent** service therein, and select the **Start** option from the shortcut menu that appears.

Once the collector service is started, the AppFlow collector will begin receiving AppFlow records from the managed NetScaler device. These records are then processed and stored by the collector in binary files. At configured frequencies, each Web AppFlow test that the eG remote agent runs, will read these binary files to pull and aggregate AppFlow statistics. The statistics so aggregated are then reported to the eG manager, which publishes these metrics in the eG monitoring console. To know what tests the remote agent runs, what metrics it collects, and how these metrics are grouped and presented in the eG monitoring console, refer to the Section **Chapter 3** topic.

Chapter 3: Monitoring NetScaler Web AppFlow

eG Enterprise provides a specialized *Citrix Web AppFlow* monitoring model for monitoring Web AppFlow.

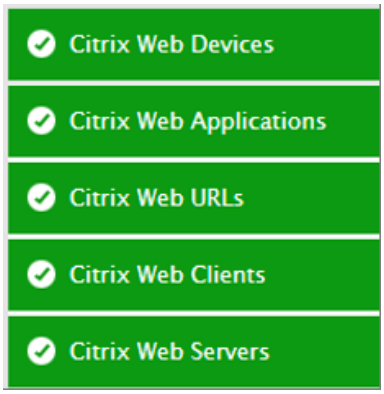


Figure 3.1: The layer model of Citrix Web AppFlow

Each layer of Figure 3.1 above is mapped to tests that intercept AppFlow records to report a wide variety of useful metrics. Using these metrics, administrators can find quick and accurate answers for the following performance queries:

- Is web traffic handled by the target NetScaler appliance abnormally high?
- Is the NetScaler appliance consuming bandwidth excessively when processing this traffic?
- Is any web application processing bandwidth-intensive traffic?
- Which is the exact web URL that was hit when bandwidth usage peaked?
- Are specific web protocols consuming more bandwidth than the rest?
- Are any HTTP request methods bandwidth-intensive?
- Is any specific HTTP response message from the web application hogging the bandwidth?
- Are operations performed by a particular web client bandwidth-intensive? If so, which web client is it? Which operating system is that client running on? Is high bandwidth usage by the web client OS impacting web client performance?
- Is any user agent/client device (mobile phone, PC, tablet) over-utilizing the bandwidth resources?
- How well is the NetScaler appliance processing web application traffic? Were any latencies noticed in the appliance?

- Which web application is responding to requests very slowly, and where is the bottleneck - at the server end? or client end?
- Is poor web application performance owing to the slow responsiveness of any particular web URL? If so, which URL is it, and where is the problem - in page loading? or in page rendering?
- Is any web protocol latent? If so, where is the protocol seeing maximum slowness - on the server-side? or on the client-side?
- Is any HTTP response status message taking too long to be rendered, thus degrading web application performance?
- Can slowness experienced by web application users be attributed to any specific web clients / user agents (i.e., devices such as mobile phones, PC, tablet, etc.) they are using? If so, which ones?
- If a latent web client is to blame for poor user experience with a web application, then is the slowness experienced by the web client owing to a slow operating system?
- Is any web virtual server processing traffic slowly?

This chapter discusses each layer of Figure 3.1 and the tests mapped to every layer.

3.1 The Citrix Web Servers Layer

Using the tests mapped to this layer, you can identify web servers and virtual servers that at poorly responsive to client requests, and isolate the reason for the slowness.

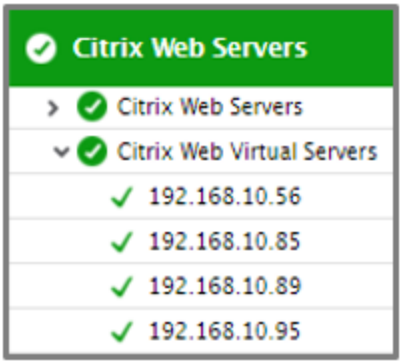


Figure 3.2: The tests mapped to the Citrix Web Servers layer

3.1.1 Citrix Web Servers Test

If users accessing a web server complain of slowness, administrators must be able to quickly figure out what is causing the slowness – is it because of a processing bottleneck with the web server? or is

it owing to a latent server network? The **Citrix Web Servers** test accurately points administrators to the source of the slowness! This test tracks requests to each web server managed by NetScaler and reports the time every server takes to process the requests. The test thus sends out proactive alerts to administrators if it finds that any web server is responding very slowly to client requests. Additionally, the test also indicates if the slowdown experienced by the user can be attributed to a latent server-side network. This way, the test helps administrators identify slow servers and rapidly isolate the reason for the slowness, so that the problem can be fixed quickly and normalcy restored in no time.

Target of the test : An AppFlow-enabled NetScaler appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every web server that hosts web applications managed by a NetScaler appliance

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling

Parameter	Description
	effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_ DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.
Show Top N Servers	By default, this is set to **Yes**. This means that, by default, the test will report metrics for only the top web servers (in terms of number of hits or bandwidth usage). In this case, only the top-N bandwidth-intensive or most-used web servers (depending upon the option chosen against the **Show Top-N Servers By** parameter) will be the descriptors of this test. If you want the test to report metrics for all web servers, then

Parameter	Description
	set this flag to **No**.
Show Top N Servers By	By default, this parameter is set to **Hits**. This means that, by default, the test will report metrics for only those web servers that have been used the most. If required, you can configure the test to report metrics for those web servers that are bandwidth-intensive. For that, set this parameter to **Bandwidth**.
Top N Servers Limit	By default, this is set to 10. This denotes that the test will report metrics for the top-10 web servers (in terms of number of hits or bandwidth usage, depending upon the **Show Top-N Servers By** parameter setting) only. You can change the 'N' in top-N by specifying a higher or a lower value here.
Show Top N in DD	By default, this flag is set to **Yes**. This indicates that, by default, the detailed diagnosis of this test will display the details of only the top requests for a server (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting). If you set this flag to **No**, then detailed diagnosis will provide the details of all requests.
Sort DD Data By	By default, this test sorts the detailed diagnostics it reports in the descending order of those HTTP request method:response status pairs that have seen the maximum hits. Accordingly, the **Hits** option is by default chosen against this parameter. Detailed diagnosis so sorted will point you to those server requests that frequently returned error responses. If required, you can sort the detailed diagnostics in the descending order of bandwidth usage, so you can quickly identify those requests that resulted in bandwidth-intensive responses. For this, choose the **Bandwidth** option against this parameter.
Top N DD Limit	**This parameter applies only if the Show Top N in DD flag is set to 'Yes'.** By default, this parameter is set to 10, indicating that the detailed diagnostics will report the top-10 HTTP request method:response status pairs (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting). You can change the 'N' in Top N by specifying any number of your choice in this text box.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against **DD Frequency**.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are

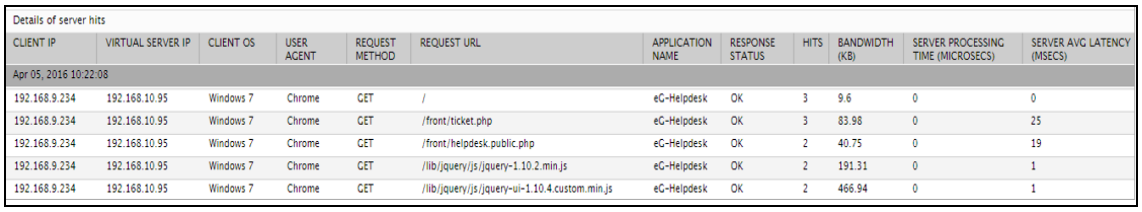
Parameter	Description
	detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests received by this web server.	Number	This is a good indicator of the load on the web server. Compare the value of this measure across web servers to know which server is receiving the maximum number of requests. If a single server appears to be servicing a significantly large number of requests than the rest, it could imply that the server is overloaded. This in turn indicates that a faulty/ineffective load-balancing algorithm is in use. Use the detailed diagnosis of this measure to identify the bandwidth-intensive requests to the web server and requests that have often failed/resulted in error responses.
Bandwidth	Indicates the total amount of data received by this web server.	KB	Compare the value of this measure across web servers to know which server is consuming bandwidth excessively.
Server processing	Indicates the elapsed	msecs	A high value for this measure indicates

Measurement	Description	Measurement Unit	Interpretation
time	time, from when the server starts to receive the first byte of a request from the NetScaler appliance until the NetScaler appliance receives the first byte to response.		that the web server is processing requests slowly. Compare the value of this measure across web servers to isolate the slowest web server. In the event that a user complains of slowness, you can compare the value of this measure with that of the *Server avg latency* measure to determine what is causing the slowness – the poor processing power of the web server? or a latent server network?
Server avg latency	Indicates the average latency caused by the server network.	msecs	A high value for this measure indicates that the server network is latent. Compare the value of this measure across web servers to know which server's network is the slowest. In the event that a user complains of slowness, you can compare the value of this measure with that of the *Server processing time* measure to determine what is causing the slowness – the poor processing power of the web server? or a latent server network?

The detailed diagnosis of the *Hits* measure groups server requests on the basis of the HTTP request method and response status of the requests. For each unique HTTP request method:response status pair, the detailed diagnosis reveals the client from which the requests were received, the OS of the client, the device used for sending the requests, and the web server to which the requests were sent. Additionally, the detailed diagnostics also report the number of hits, bandwidth usage, and responsiveness of each HTTP request method:response status pair. In the process, the test points to request methods that often resulted in error responses, request methods that took too long to be serviced, and the probable cause for the poor responsiveness - did the server take too long to process requests of that type? or is the slowness owing to a latent server network?



Details of server hits

CLIENT IP	VIRTUAL SERVER IP	CLIENT OS	USER AGENT	REQUEST METHOD	REQUEST URL	APPLICATION NAME	RESPONSE STATUS	HITS	BANDWIDTH (KB)	SERVER PROCESSING TIME (MICROSECS)	SERVER AVG LATENCY (MSECS)
Apr 05, 2016 10:22:08											
192.168.9.234	192.168.10.95	Windows 7	Chrome	GET	/	eG-Helpdesk	OK	3	9.6	0	0
192.168.9.234	192.168.10.95	Windows 7	Chrome	GET	/front/ticket.php	eG-Helpdesk	OK	3	83.98	0	25
192.168.9.234	192.168.10.95	Windows 7	Chrome	GET	/front/helpdesk.public.php	eG-Helpdesk	OK	2	40.75	0	19
192.168.9.234	192.168.10.95	Windows 7	Chrome	GET	/lib/jquery/js/jquery-1.10.2.min.js	eG-Helpdesk	OK	2	191.31	0	1
192.168.9.234	192.168.10.95	Windows 7	Chrome	GET	/lib/jquery/js/jquery-ui-1.10.4.custom.min.js	eG-Helpdesk	OK	2	466.94	0	1

Figure 3.3: The detailed diagnosis of the Hits measure reported by the Citrix Web Servers test

3.1.2 Citrix Web Virtual Servers Test

This test tracks requests to each virtual server managed by NetScaler and reports the time every server takes to process the requests. The test thus sends out proactive alerts to administrators if it finds that any virtual server is responding very slowly to client requests. Additionally, the test also indicates if the slowdown experienced by the user can be attributed to a latent client-side network. This way, the test helps administrators identify slow virtual servers and rapidly isolate the reason for the slowness, so that the problem can be fixed quickly and normalcy restored in no time.

Target of the test : An AppFlow-enabled NetScaler appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every virtual server managed by the monitored NetScaler appliance

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster

Parameter	Description
	node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly

Parameter	Description
	created, then decrease the value of this parameter. 7. Finally, save the file.
Show Top N VIP	By default, this is set to **Yes**. This means that, by default, the test will report metrics for only the top virtual servers (in terms of number of hits or bandwidth usage). In this case, only the top-N bandwidth-intensive or most-used virtual servers (depending upon the option chosen against the **Show Top-N VIP By** parameter) will be the descriptors of this test. If you want the test to report metrics for all virtual servers, then set this flag to **No**.
Show Top N VIP By	By default, this parameter is set to **Hits**. This means that, by default, the test will report metrics for only those virtual servers that have received the maximum number of requests. If required, you can configure the test to report metrics for those virtual servers that are bandwidth-intensive. For that, set this parameter to **Bandwidth**.
Top N Servers Limit	By default, this is set to 10. This denotes that the test will report metrics for the top-10 virtual servers (in terms of number of hits or bandwidth usage, depending upon the **Show Top-N VIP By** parameter setting) only. You can change the 'N' in top-N by specifying a higher or a lower value here.
Show Top N in DD	By default, this flag is set to **Yes**. This indicates that, by default, the detailed diagnosis of this test will display the details of only the top requests to a virtual server (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting). If you set this flag to **No**, then detailed diagnosis will provide the details of all requests.
Sort DD Data By	By default, this test sorts the detailed diagnostics it reports in the descending order of those HTTP request method:response status pairs that have seen the maximum hits. Accordingly, the **Hits** option is by default chosen against this parameter. Detailed diagnosis so sorted will point you to those requests that frequently returned error responses. If required, you can sort the detailed diagnostics in the descending order of bandwidth usage, so you can quickly identify those requests that resulted in bandwidth-intensive responses. For this, choose the **Bandwidth** option against this parameter.
Top N DD Limit	**This parameter applies only if the Show Top N in DD flag is set to 'Yes'.** By default, this parameter is set to 10, indicating that the detailed diagnostics will report the top-10 HTTP request method:response status pairs (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting). You can change the 'N' in Top N by specifying any number of your choice in this text box.

Parameter	Description
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against **DD Frequency**.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests received by this virtual server.	Number	This is a good indicator of the load on the virtual server. Compare the value of this measure across virtual servers to know which server is receiving the maximum number of requests. If a single server appears to be servicing a significantly large number of requests than the rest, it could imply that the server is overloaded. This in turn indicates that a faulty/ineffective load-balancing algorithm is in use. Use the detailed diagnosis of this measure to identify the bandwidth-intensive requests to the server and

Measurement	Description	Measurement Unit	Interpretation
			requests that have often failed/resulted in error responses.
Bandwidth	Indicates the total amount of data received by this virtual server.	KB	Compare the value of this measure across virtual servers to know which server is consuming bandwidth excessively.
VIP avg latency	Indicates the latency caused by the client-side network.	msecs	A high value for this measure is a cause for concern as it indicates that a bottleneck in the client-side network.
VIP max latency	Indicates the high watermark of client network latency.	msecs	

The detailed diagnosis of the *Hits* measure groups requests to a virtual server on the basis of the HTTP request method and response status of the requests. For each unique HTTP request method:response status pair, the detailed diagnosis reveals the client from which the requests were received, the OS of the client, the device used for sending the requests, and the web server to which the requests were sent. Additionally, the detailed diagnostics also report the number of hits, bandwidth usage, and responsiveness of each HTTP request method:response status pair. In the process, the test points to request methods that often resulted in error responses and request methods that were delayed due to the client-side network.

Details of virtual servers

CLIENT IP	SERVER IP	CLIENT OS	USER AGENT	REQUEST METHOD	REQUEST URL	APPLICATION NAME	RESPONSE STATUS	HITS	BANDWIDTH (KB)	RENDER TIME (MSECS)	VIP AVG LATENCY (MSECS)
Apr 05, 2016 10:22:20											
192.168.9.234	-	Windows 7	Chrome	GET	/plugins/news/scripts/alert.php	eG-Helpdesk	-	4	0	0	1
192.168.9.234	192.168.8.67	Windows 7	Chrome	GET	/front/ticket.php	eG-Helpdesk	OK	3	83.98	0	7
192.168.9.234	192.168.8.67	Windows 7	Chrome	GET	/	eG-Helpdesk	OK	3	9.6	0	2
192.168.9.234	192.168.8.67	Windows 7	Chrome	GET	/lib/jquery/js/jquery-ui-1.10.4.custom.min.js	eG-Helpdesk	OK	2	466.94	0	13
192.168.9.234	192.168.8.67	Windows 7	Chrome	GET	/front/helpdesk.public.php	eG-Helpdesk	OK	2	40.75	0	7

Figure 3.4: The detailed diagnosis of the Hits measure reported by the Citrix Web Virtual Servers test

3.2 The Citrix Web Clients Layer

The test mapped to this layer helps administrators determine why users from a particular web client alone are complaining of slowness - is it because of a bandwidth contention? a sluggish client network? or poor page rendering capacity?

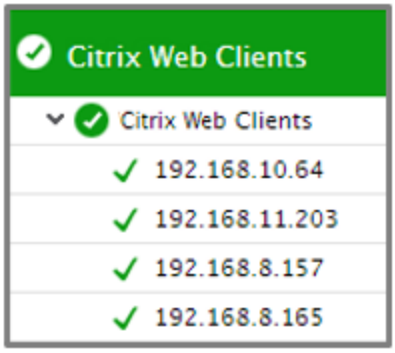


Figure 3.5: The test mapped to the Citrix Web Clients layer

3.2.1 Citrix Web Clients Test

When encountered by a request overload on a web server, administrators must quickly identify the client that could have contributed to that load. Also, when users connecting from a specific client complain of slowness, administrators should be able to swiftly zero-in on the root-cause of that slowness, so that the problem can be resolved before user productivity is impacted. The **Citrix Web Clients** test helps in both accounts! This test tracks requests from every client connecting to the web servers managed by a NetScaler appliance, and reports the requests count, bandwidth usage, page rendering time, and the network latency for each client. From these metrics, administrators can easily infer which client is imposing the maximum load on the web servers. Moreover, if help desk receives frequent complaints of slowness from users connecting from a particular client, then administrators can use the metrics reported by this test to isolate the source of the delay – is it because the client is unable to render the response pages quickly? is it because of a latent client network? or is it owing to a contention for bandwidth resources?

Target of the test : An AppFlow-enabled NetScaler appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every web client that connects to the web applications managed by the target NetScaler appliance

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**

Parameter	Description
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can

Parameter	Description
	increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.
Show Top N Clients	By default, this is set to **Yes**. This means that, by default, the test will report metrics for only the top clients (in terms of number of hits or bandwidth usage). In this case, only the top-N bandwidth-intensive or most-accessed clients (depending upon the option chosen against the **Show Top-N Clients By** parameter) will be the descriptors of this test. If you want the test to report metrics for all URLs, then set this flag to **No**.
Show Top N Clients By	By default, this parameter is set to **Hits**. This means that, by default, the test will report metrics for only those web clients that have received the maximum number of hits. If required, you can configure the test to report metrics for those clients that are bandwidth-intensive. For that, set this parameter to **Bandwidth**.
Top N Clients Limit	By default, this is set to 10. This denotes that the test will report metrics for the top-10 web clients (in terms of number of hits or bandwidth usage, depending upon the **Show Top-N Clients By** parameter setting) only. You can change the 'N' in top-N by specifying a higher or a lower value here.
Show Top N in DD	By default, this flag is set to **Yes**. This indicates that the detailed diagnosis of this test will display the details of only the top requests for the web client (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting), by default. If you set this flag to **No**, then detailed diagnosis will provide the details of all URLs.
Sort DD Data By	By default, this test sorts the detailed diagnostics it reports in the descending order of those HTTP request method:response status pairs that have seen the maximum hits. Accordingly, the **Hits** option is by default chosen against this parameter. Detailed diagnosis so sorted will point you to those requests that frequently returned error responses. If required, you can sort the detailed diagnostics in the descending order

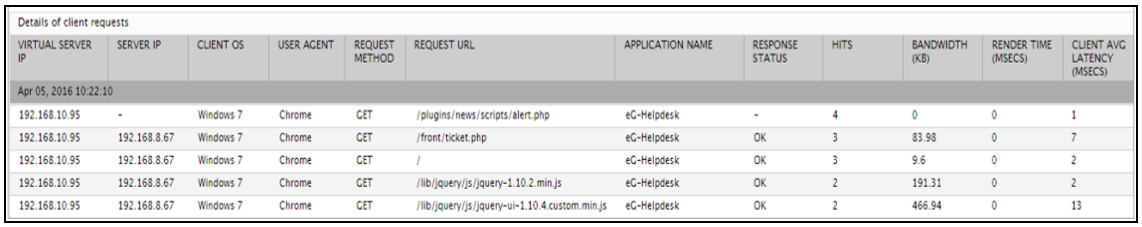
Parameter	Description
	of bandwidth usage, so you can quickly identify those application requests that resulted in bandwidth-intensive responses. For this, choose the **Bandwidth** option against this parameter.
Top N DD Limit	**This parameter applies only if the Show Top N in DD flag is set to 'Yes'.** By default, this parameter is set to 10, indicating that the detailed diagnostics will report the top-10 HTTP request method:response status pairs (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting). You can change the 'N' in Top N by specifying any number of your choice in this text box.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against **DD Frequency**.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Requests	Indicates the number of requests received from this client.	Number	In the event of an overload condition, you can compare the value of this measure across clients to know which client is overloading the servers. Use the detailed diagnosis of this measure to know which servers this

Measurement	Description	Measurement Unit	Interpretation
			client is accessing and which NetScaler manages the web traffic generated by this client.
Bandwidth	Indicates the total amount of data received from this client.	KB	Compare the value of this measure across clients to know which client is hogging the bandwidth. In the event of a slowdown, you can use the value of this measure to figure out if the lack of adequate bandwidth is what is slowing down user accesses from this client.
Avg render time	Indicates the elapsed time, from when the browser starts to receive the first byte of a response until either all page content has been rendered by this client or the page load action has timed out.	msecs	A high value for this measure is a cause for concern as it indicates that the client is delaying page rendering. In the event of a slowdown, you can compare the value of this measure with that of the *Client avg latency* measure to zero-in on the root-cause of the slowdown - is it because the client is unable to render the response pages quickly? or is it because of a latent client network?
Client avg latency	Indicates the latency caused by the client-side network.	msecs	A high value for this measure is a cause for concern as it indicates that a bottleneck in the client-side network. In the event of a slowdown, you can compare the value of this measure with that of the *Avg render time* measure to zero-in on the root-cause of the slowdown - is it because the client is unable to render the response pages quickly? or is it because of a latent client network?
Client max latency	Indicates the high watermark of client network latency.	msecs	

The detailed diagnosis of the *Requests* measure groups application requests on the basis of the HTTP request method and response status of the requests. For each unique HTTP request method:response status pair, the detailed diagnosis reveals the client from which the requests were received, the OS of the client, the device used for sending the requests, and the web server to which the requests were sent. Additionally, the detailed diagnostics also report the number of hits, bandwidth usage, the page render time, and the client network latency of each HTTP request method:response status pair. In the process, the test points to request methods that often resulted in error responses. Also, if applications poorly respond to requests from a particular web client, these diagnostics help isolate the probable cause for the slowness - is it because the web client takes too long to render web pages returned by the applications? or is it because of a delay in the client -side network?



Details of client requests

VIRTUAL SERVER IP	SERVER IP	CLIENT OS	USER AGENT	REQUEST METHOD	REQUEST URL	APPLICATION NAME	RESPONSE STATUS	HITS	BANDWIDTH (KB)	RENDER TIME (MSECS)	CLIENT AVG LATENCY (MSECS)
Apr 05, 2016 10:22:10											
192.168.10.95	-	Windows 7	Chrome	GET	/plugins/news/scripts/alert.php	eG-Helpdesk	-	4	0	0	1
192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	/front/ticket.php	eG-Helpdesk	OK	3	83.98	0	7
192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	/	eG-Helpdesk	OK	3	9.6	0	2
192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	/lib/jquery/js/jquery-1.10.2.min.js	eG-Helpdesk	OK	2	191.31	0	2
192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	/lib/jquery/js/jquery-ui-1.10.4.custom.min.js	eG-Helpdesk	OK	2	466.94	0	13

Figure 3.6: The detailed diagnosis of the Requests measure reported by the Citrix Web Clients test

3.3 The Citrix Web URLs Layer

With the help of the test mapped to this layer, administrators can identify the URLs that are responsible for bandwidth-intensive traffic, the URLs that are responding slowly to requests, and the reason for the slowness.

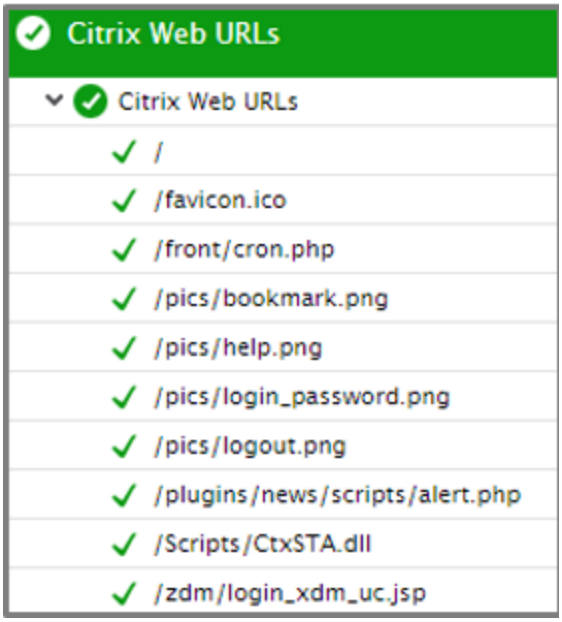


Figure 3.7: The test mapped to the Citrix Web URLs layer

3.3.1 Citrix Web URLs Test

If traffic to a web application is found to be bandwidth-intensive, administrators may instantly want to know which specific URL in that application is actually hogging the bandwidth resources. Administrators may later want to block access to these URLs, so as to conserve bandwidth. Likewise, if user experience with a web application deteriorates, administrators may want to swiftly check the responsiveness of each URL requested by the user, to identify which URL is adversely impacting the user experience and why. The **Citrix Web URLs** test helps administrators with this. This test automatically discovers the URLs accessed by users and reports the number of requests received and bandwidth used per URL. This way, the test pinpoints those URLs that are accessed frequently and the ones that generate bandwidth-intensive web traffic. Additionally, the test also reports the page load time and render time of every URL, so that, when a slowdown occurs, administrators can instantly identify the URL that resulted in a slow response and where the slowdown occurred – when the requested page was loaded? Or when it was rendered by the client?

Target of the test : An AppFlow-enabled NetScaler appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every URL that was accessed

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**.

Parameter	Description
	4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.
Show Top N URLs	By default, this is set to **Yes**. This means that, by default, the test will report metrics for only the top URLs (in terms of number of hits or bandwidth usage). In this case, only the top-N bandwidth-intensive or most-accessed URLs (depending upon the option chosen against the **Show Top N URLs By** parameter) will be the descriptors of this test. If you want the test to report metrics for all URLs, then set this flag to **No**.
Show Top N URLs By	By default, this parameter is set to **Hits**. This means that, by default, the test will report metrics for only those URLs that have received the maximum number of hits. If required, you can configure the test to report metrics for those URLs that are bandwidth-intensive. For that, set this parameter to **Bandwidth**.
Top N URLs Limit	By default, this is set to 10. This denotes that the test will report metrics for the top-10 URLs (in terms of number of hits or bandwidth usage, depending upon the **Show Top N URLs By** parameter setting) only. You can change the 'N' in top-N by specifying a higher or a lower value here.
Show Top N in DD	By default, this flag is set to **Yes**. This indicates that the detailed diagnosis of this test will display the details of only the top requests for the URL (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting), by default. If you set this flag to **No**, then detailed diagnosis will provide the details of all URLs.
Sort DD Data By	By default, this test sorts the detailed diagnostics it reports in the descending order of those HTTP request method:response status pairs that have seen the maximum hits.

Parameter	Description
	Accordingly, the **Hits** option is by default chosen against this parameter. Detailed diagnosis so sorted will point you to those requests that frequently returned error responses. If required, you can sort the detailed diagnostics in the descending order of bandwidth usage, so you can quickly identify those application requests that resulted in bandwidth-intensive responses. For this, choose the **Bandwidth** option against this parameter.
Top N DD Limit	**This parameter applies only if the Show Top N in DD flag is set to 'Yes'.** By default, this parameter is set to 10, indicating that the detailed diagnostics will report the top-10 HTTP request method:response status pairs (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting). You can change the 'N' in Top N by specifying any number of your choice in this text box.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against **DD Frequency**.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests received by this web URL.	Number	This is a good indicator of how popular a URL is. Compare the value of this measure

Measurement	Description	Measurement Unit	Interpretation
			across web URLs to know which application is the most popular. Use the detailed diagnosis of this measure to identify the bandwidth-intensive requests to the URL and requests that have often failed/resulted in error responses.
Bandwidth	Indicates the total amount of data received by this web URL.	KB	Compare the value of this measure across URLs to know which URL is consuming bandwidth excessively.
Avg load time	Indicates the elapsed time, from when the browser starts to receive the first byte of a response until the user starts to interact with the page. At this stage, some of the page content might not yet have been loaded.	msecs	A high value for this measure is a cause for concern as it indicates that the requested web page is taking too long to load. In the event of a slowdown, you may want to compare the value of this measure with that of the *Avg render time* measure to accurately determine the reason for the slowness – is it because of a delay in page loading? Or page rendering?
Avg render time	Indicates the elapsed time, from when the browser starts to receive the first byte of a response until either all page content has been rendered or the page load action has timed out.	msecs	A high value for this measure indicates that the client is having problems rendering the requested pages. In the event of a slowdown, you may want to compare the value of this measure with that of the *Avg load time* measure to accurately determine the reason for the slowness – is it because of a delay in page loading? Or page rendering?

The detailed diagnosis of the *Hits* measure groups application requests on the basis of the HTTP request method and response status of the requests. For each unique HTTP request method:response status pair, the detailed diagnosis reveals the client from which the requests were received, the OS of the client, the device used for sending the requests, and the web server to which the requests were sent. Additionally, the detailed diagnostics also report the number of hits,

bandwidth usage, and responsiveness of each HTTP request method:response status pair. In the process, the test points to request methods that often resulted in error responses. Additionally, these detailed metrics will help you identify times when the web page (URL) requested took too long to load or was too slow to be rendered.

Details of web URL hits

CLIENT IP	VIRTUAL SERVER IP	SERVER IP	CLIENT OS	USER AGENT	REQUEST METHOD	APPLICATION NAME	RESPONSE STATUS	HITS	BANDWIDTH (KB)	RENDER TIME (MSECS)	LOAD TIME (MSECS)
Apr 04, 2016 15:51:09											
192.168.8.241	192.168.10.95	192.168.8.67	Windows 7	Firefox	POST	eG-Helpdesk	OK	1	1.7	0	0

Figure 3.8: The detailed diagnosis of the Hits measure reported by the Citrix Web URLs test

3.4 The Citrix Web Applications Layer

The tests mapped to this layer help analyze web traffic at the protocol- and web application-levels. In the process, the tests pinpoint:

- Bandwidth-hungry protocols, slow protocols, and the probable cause for the slowness;
- Bandwidth-hungry web applications, least responsive web applications, and the source of the application slowness

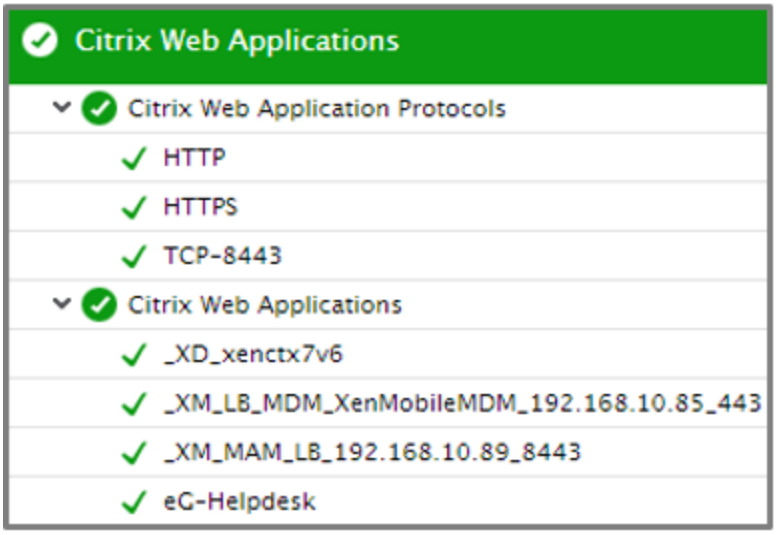


Figure 3.9: The tests mapped to the Citrix Web Applications layer

3.4.1 Citrix Web Application Protocols Test

Certain web protocols may require more processing power and time than others. When applications are accessed using such protocols, application performance is sure to be affected. To ensure peak application performance at all times, administrators need to quickly identify resource-intensive and

least responsive protocols, and configure server size and firewall policies accordingly. This is where the Citrix Web Application Protocols test helps!

This test auto-discovers the web protocols used for accessing the web applications managed by the monitored NetScaler appliance. For each web protocol, the test reports the bandwidth usage and responsiveness of that web protocol, thus pointing administrators to bandwidth-intensive protocols and those that respond poorly to requests. For a slow protocol, the test additionally pinpoints the probable cause of the slowness - is the server hosting the application not configured right to support that protocol? or is the client or server network slow in processing that protocol?

Target of the test : An AppFlow-enabled NetScaler appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every web application protocol

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting.

Parameter	Description
	Note: By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_ DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.
Show Top N Protocols	By default, this is set to **Yes**. This means that, by default, the test will report metrics for only the top protocols (in terms of number of hits or bandwidth usage). In this case, only the top-N bandwidth-intensive or most-used protocols (depending upon the option chosen against the **Show Top-N Protocols By** parameter) will be the descriptors of this test. If you want the test to report metrics for all protocols, then set this flag to **No**.

Parameter	Description
Show Top N Protocols By	By default, this parameter is set to **Hits**. This means that, by default, the test will report metrics for only those protocols that have been used the most. If required, you can configure the test to report metrics for those protocols that are bandwidth-intensive. For that, set this parameter to **Bandwidth**.
Top N Protocols Limit	By default, this is set to 10. This denotes that the test will report metrics for the top-10 protocols (in terms of number of hits or bandwidth usage, depending upon the **Show Top-N Protocols By** parameter setting) only. You can change the 'N' in top-N by specifying a higher or a lower value here.
Show Top N in DD	By default, this flag is set to **Yes**. This indicates that, by default, the detailed diagnosis of this test will display the details of only the top requests for a protocol (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting). If you set this flag to **No**, then detailed diagnosis will provide the details of all requests
Sort DD Data By	By default, this test sorts the detailed diagnostics it reports in the descending order of those HTTP request method:response status pairs that have seen the maximum hits. Accordingly, the **Hits** option is by default chosen against this parameter. Detailed diagnosis so sorted will point you to those requests that frequently returned error responses. If required, you can sort the detailed diagnostics in the descending order of bandwidth usage, so you can quickly identify those requests that resulted in bandwidth-intensive responses. For this, choose the **Bandwidth** option against this parameter.
Top N DD Limit	**This parameter applies only if the Show Top N in DD flag is set to 'Yes'.** By default, this parameter is set to 10, indicating that the detailed diagnostics will report the top-10 HTTP request method:response status pairs (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting). You can change the 'N' in Top N by specifying any number of your choice in this text box.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against **DD Frequency**.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular

Parameter	Description
	server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests to web applications that used this protocol.	Number	This is a good indicator of the load imposed by a protocol on web applications. Compare the value of this measure across protocols to know which protocol is most used. Use the detailed diagnosis of this measure to identify the bandwidth-intensive requests to the application that used a particular protocol, and requests that have often failed/resulted in error responses.
Bandwidth	Indicates the total amount of data used by this protocol.	KB	Compare the value of this measure across protocols to know which protocol is consuming bandwidth excessively.
Response time	Indicates the elapsed time between the end of a request that used this protocol and the beginning of a response from the application.	msecs	A high value for this measure indicates that the application is processing requests that use this protocol, slowly. If this measure reports an abnormally high value for a protocol, compare the value of the *Server processing time, Client avg latency,* and *Server avg*

Measurement	Description	Measurement Unit	Interpretation
			latency measures of that protocol to determine the reason for the slowness.
Server processing time	Indicates the elapsed time, from when the server starts to receive the first byte of a request that used this protocol until the NetScaler appliance receives the first byte to response.	msecs	A high value for this measure indicates that the web server is processing requests that used this protocol, slowly. If the *Response time* measure reports an abnormally high value for a protocol, then compare the value of this measure with that of the *Client avg latency* and *Server avg latency* measures of that protocol to determine what is causing the slowness – the poor processing power of the web server? a latent server network? or a slow client network?
Server avg latency	Indicates the average latency caused by the server network.	msecs	A high value for this measure indicates that the server network is latent. If the *Response time* measure reports an abnormally high value for a protocol, then compare the value of this measure with that of the *Client avg latency* and *Server processing time* measures of that protocol to determine what is causing the slowness – the poor processing power of the web server? a latent server network? or a slow client network?
Server max latency	Indicates the high watermark of server network latency.	msecs	
Client avg latency	Indicates the average latency caused by the client network.	msecs	A high value for this measure indicates that the client network is latent. If the *Response time* measure reports an abnormally high value for a protocol, then compare the value of

Measurement	Description	Measurement Unit	Interpretation
			this measure with that of the *Server avg latency* and *Server processing time* measures of that protocol to determine what is causing the slowness – the poor processing power of the web server? a latent server network? or a slow client network?
Client max latency	Indicates the high watermark of client network latency.	msecs	

The detailed diagnosis of the *Hits* measure groups application requests on the basis of the HTTP request method and response status of the requests. For each unique HTTP request method:response status pair, the detailed diagnosis reveals the client from which the requests were received, the OS of the client, the device used for sending the requests, and the web server to which the requests were sent. Additionally, the detailed diagnostics also report the number of hits, bandwidth usage, and responsiveness of each HTTP request method:response status pair. In the process, the test points to request methods that often resulted in error responses, request methods that took too long to be serviced by the application, and the probable cause for the poor responsiveness - did the server hosting the application take too long to process requests of that type? or is the slowness owing to a latent client or server network?

Details of Application Protocols

CLIENT IP	VIRTUAL SERVER IP	SERVER IP	CLIENT OS	USER AGENT	REQUEST METHOD	APPLICATION NAME	RESPONSE STATUS	HITS	BANDWIDTH (KB)	RESPONSE TIME (MSECS)	SERVER PROCESSIN TIME (MICROSEC	CLIENT AVG LATENCY (MSECS)	SERVER AVG LATENCY (MSECS)
Apr 04, 2016 18:51:01													
192.168.9.234	192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	eG-Helpdesk	Not Modified	34	28.72	18	0	18	1
192.168.9.234	192.168.10.95	-	Windows 7	Chrome	GET	eG-Helpdesk	-	23	0	33	0	33	0
192.168.9.234	192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	eG-Helpdesk	OK	12	816.1	7	0	7	15
192.168.9.234	192.168.10.95	192.168.8.67	Windows 7	Chrome	POST	eG-Helpdesk	OK	1	1.64	30	0	30	7

Figure 3.10: The detailed diagnosis of the Hits measure reported by the Citrix Web Application Protocols test

3.4.2 Citrix Web Applications Test

One of the key value propositions of the NetScaler appliance is its ability to optimize user experience by making web applications run faster and by delivering them to users more quickly. If users continue to complain of slowness when accessing applications despite the usage of NetScaler , it will beat the very purpose of the NetScaler, forcing enterprises to hunt for alternatives. To avoid such an eventuality, administrators should be able to detect a current/potential slowdown in a web

application well before users notice, isolate the source of the slowness, and address it rapidly. This is where the **Citrix Web Applications** Test helps! This test monitors how responsive each web application delivered by the NetScaler appliance is to user requests and promptly notifies administrators if any application's responsiveness dips. Upon receiving such intimations, administrators can once again zoom into the test to figure out the reason for the slowness- is it because the web server hosting the application is unable to process requests quickly? is it owing to a latent server-side network? or is it due to a latent client-side network? Additionally, the test also monitors the bandwidth usage of each web application, thus pointing administrators to bandwidth-intensive applications.

Target of the test : An AppFlow-enabled NetScaler appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every web application managed by the monitored NetScaler appliance

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_ AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_ NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files

Parameter	Description
	provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_ DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.
Show Top N Applications	By default, this is set to **Yes**. This means that, by default, the test will report metrics for only the top applications (in terms of number of hits or bandwidth usage). In this case, only the top-N bandwidth-intensive or most-accessed applications (depending upon the option chosen against the **Show Top-N Applications By** parameter) will be

Parameter	Description
	the descriptors of this test. If you want the test to report metrics for all applications, then set this flag to **No**.
Show Top N Applications By	By default, this parameter is set to **Hits**. This means that, by default, the test will report metrics for only those applications that have received the maximum number of hits. If required, you can configure the test to report metrics for those applications that are bandwidth-intensive. For that, set this parameter to **Bandwidth**.
Top N Applications Limit	By default, this is set to 10. This denotes that the test will report metrics for the top-10 applications (in terms of number of hits or bandwidth usage, depending upon the **Show Top-N Applications By** parameter setting) only. You can change the 'N' in top-N by specifying a higher or a lower value here.
Show Top N in DD	By default, this flag is set to **Yes**. This indicates that the detailed diagnosis of this test will display the details of only the top requests for an application (in terms of the number of hits or bandwidth usage, depending upon the **Sort DD Data By** setting), by default. If you set this flag to **No**, then detailed diagnosis will provide the details of all requests to a web application.
Sort DD Data By	By default, this test sorts the detailed diagnostics it reports in the descending order of those HTTP request method:response status pairs that have seen the maximum hits. Accordingly, the **Hits** option is by default chosen against this parameter. Detailed diagnosis so sorted will point you to those application requests that frequently returned error responses. If required, you can sort the detailed diagnostics in the descending order of bandwidth usage, so you can quickly identify those application requests that resulted in bandwidth-intensive responses. For this, choose the **Bandwidth** option against this parameter.
Top N DD Limit	**This parameter applies only if the Show Top N in DD flag is set to 'Yes'.** By default, this parameter is set to 10, indicating that the detailed diagnostics will report the top-10 HTTP request method:response status pairs (in terms of the number of hits or bandwidth usage, depending upon the Sort DD Data By setting). You can change the 'N' in Top N by specifying any number of your choice in this text box.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against **DD Frequency**.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be

Parameter	Description
	configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests received by this web application.	Number	This is a good indicator of the load on the web application. Compare the value of this measure across web applications to know which application is the most popular. Use the detailed diagnosis of this measure to identify the bandwidth-intensive requests to the application and requests that have often failed/resulted in error responses.
Bandwidth	Indicates the total amount of data received by this web application.	KB	Compare the value of this measure across web applications to know which web application is consuming bandwidth excessively.
Response time	Indicates the elapsed time between the end of an enquiry and the beginning of a response from this application.	msecs	A high value for this measure indicates that the application is processing requests slowly. If this measure reports an abnormally high value for an application, compare the value of the *Server processing time, Client avg latency,* and *Server avg latency* measures of that

Measurement	Description	Measurement Unit	Interpretation
			application to determine the reason for the slowness.
Server processing time	Indicates the elapsed time, from when the server starts to receive the first byte of a request from the NetScaler appliance until the NetScaler appliance receives the first byte to response.	msecs	A high value for this measure indicates that the web server is processing requests slowly. If the *Response time* measure reports an abnormally high value for a web application, then compare the value of this measure with that of the *Client avg latency* and *Server avg latency* measures of that application to determine what is causing the slowness – the poor processing power of the web server? a latent server network? or a slow client network?
Server avg latency	Indicates the average latency caused by the server network.	msecs	A high value for this measure indicates that the server network is latent. If the *Response time* measure reports an abnormally high value for a web application, then compare the value of this measure with that of the *Client avg latency* and *Server processing time* measures of that application to determine what is causing the slowness – the poor processing power of the web server? a latent server network? or a slow client network?
Server max latency	Indicates the high watermark of server network latency.	msecs	
Client avg latency	Indicates the average latency caused by the client network.	msecs	A high value for this measure indicates that the client network is latent. If the *Response time* measure reports an abnormally high value for a web application, then compare the

Measurement	Description	Measurement Unit	Interpretation
			value of this measure with that of the *Server avg latency* and *Server processing time* measures of that application to determine what is causing the slowness – the poor processing power of the web server? a latent server network? or a slow client network?
Client max latency	Indicates the high watermark of client network latency.	msecs	

The detailed diagnosis of the *Hits* measure groups application requests on the basis of the HTTP request method and response status of the requests. For each unique HTTP request method:response status pair, the detailed diagnosis reveals the client from which the requests were received, the OS of the client, the device used for sending the requests, and the web server to which the requests were sent. Additionally, the detailed diagnostics also report the number of hits, bandwidth usage, and responsiveness of each HTTP request method:response status pair. In the process, the test points to request methods that often resulted in error responses, request methods that took too long to be serviced by the application, and the probable cause for the poor responsiveness - did the server hosting the application take too long to process requests of that type? or is the slowness owing to a latent client or server network?

Details of Application hits

CLIENT IP	VIRTUAL SERVER IP	SERVER IP	CLIENT OS	USER AGENT	REQUEST METHOD	RESPONSE STATUS	HITS	BANDWIDTH (KB)	RESPONSE TIME (MSECS)	SERVER PROCESSING TIME (MICROSECS)	CLIENT AVG LATENCY (MSECS)	SERVER AVG LATENCY (MSECS)
Apr 04, 2016 18:51:15												
192.168.9.234	192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	Not Modified	34	28.72	18	0	18	1
192.168.9.234	192.168.10.95	-	Windows 7	Chrome	GET	-	23	0	33	0	33	0
192.168.9.234	192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	OK	12	616.1	7	0	7	15
192.168.9.234	192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	Not Found	1	0.99	201	0	201	1
192.168.9.234	192.168.10.95	192.168.8.67	Windows 7	Chrome	POST	OK	1	1.64	30	0	30	7

Figure 3.11: The detailed diagnosis of the Hits measure reported by the Citrix Web Applications test

3.5 The Citrix Web Devices Layer

Using the tests mapped to this layer, administrators can:

- Figure out whether/not the target NetScaler appliance is sized with adequate bandwidth resources for processing the web traffic to it;

- Identify bandwidth-intensive HTTP Request Methods and Response status;
- Pinpoint bandwidth-hungry web operating systems and client devices ;
- Spot web operating systems and client devices that are slow in rendering web pages

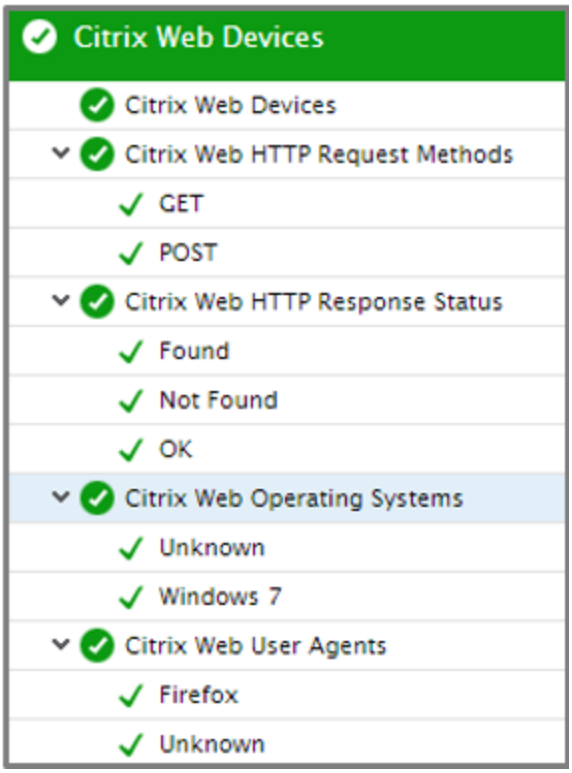


Figure 3.12: The tests mapped to the Citrix Web Devices layer

3.5.1 Citrix Web Devices Test

One of the most important tasks of an administrator is to ensure that their business-critical web applications are not impacted by any slowness or network resource contention on the NetScaler appliance managing those applications. For that purpose, administrators should periodically monitor the load on the appliance, measure its bandwidth usage, and determine whether the appliance has adequate bandwidth resources to service its load. Additionally, the administrators should also keep an eye out for latencies in NetScaler operations, so that they can proactively detect potential processing bottlenecks on the appliance and avert it. To achieve this, administrators can use the Citrix Web Devices test.

This test monitors the bandwidth usage of the NetScaler appliance and alerts administrators to bandwidth contentions. Alongside resource usage, the test also tracks the requests to the appliance, so administrators can figure out if the appliance is configured with sufficient bandwidth resources to

process the requests. Moreover, the test also captures and reports latencies experienced by the NetScaler appliance when processing the requests, thus bringing to light current/potential processing bottlenecks.

Target of the test : An AppFlow-enabled NetScaler Appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for the NetScaler appliance being monitored

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this

Parameter	Description
	default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_ DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests received by the monitored NetScaler appliance.	Number	This is a good indicator of the load on the appliance.
Bandwidth	Indicates the total amount of data received by this appliance.	KB	An abnormally high value for this measure is indicative of excessive bandwidth usage

Measurement	Description	Measurement Unit	Interpretation
NetScaler avg latency	Indicates the average latency experienced by the NetScaler appliance.	Msecs	A consistent increase in the value of this measure is a cause for concern, as it indicates a probable processing bottleneck on the appliance.
NetScaler max latency	Reports the high watermark of latency on the NetScaler appliance.	Msecs	

3.5.2 Citrix Web HTTP Request Methods

Using the **Citrix Web Devices** test, administrators can detect a bandwidth contention on the monitored NetScaler appliance. To further investigate this anomaly, administrators may want to drill down to the individual request methods used in the HTTP requests to the appliance and understand how much bandwidth each method consumes. These method-level insights can help isolate the exact method that could be eating into the bandwidth resources available to the NetScaler. The **Citrix Web HTTP Request Methods** test provides these useful method-level insights. This test auto-discovers the HTTP request methods in use and reports the number of requests received and bandwidth used per method. In the process, the test points to the most popular and the most bandwidth-intensive HTTP request methods.

Target of the test : An AppFlow-enabled NetScaler Appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every HTTP request method in use

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow

Parameter	Description
	Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be

Parameter	Description
	written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests received by the NetScaler appliance using this HTTP request method.	Number	Compare the value of this measure across methods to know which method is the most popular.
Bandwidth	Indicates the total amount of data received by the appliance using this HTTP request method.	KB	Compare the value of this measure across methods to know which HTTP request method is consuming bandwidth excessively.

3.5.3 Citrix Web HTTP Response Status Test

Like the HTTP request methods, one/more HTTP response status messages may also consume considerable bandwidth, thus increasing the bandwidth usage of the NetScaler appliances managing the web traffic. To identify such bandwith-intensive response statuses, use this test.

Target of the test : An AppFlow-enabled NetScaler Appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every HTTP response status

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of**

Parameter	Description
	capturing and aggregating AppFlow data related to the last 5 minutes only.
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you

Parameter	Description
	want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.
Show Top N in DD	By default, this flag is set to **Yes**. This indicates that the detailed diagnosis of this test will display the details of only the top responses (in terms of the number of hits or bandwidth usage), by default. If you set this flag to **No**, then detailed diagnosis will provide the details of all responses processed by the monitored NetScaler appliance for the web applications it manages.
Sort DD Data By	By default, this test sorts the detailed diagnostics it reports in the descending order of the count of HTTP responses of a type (i.e., response code) from a web application. Accordingly, the **Hits** option is by default chosen against this parameter. Detailed diagnosis so sorted will point you to web applications that sent too many error responses, so you can investigate why the web applications sent such responses. If required, you can sort the detailed diagnostics in the descending order of bandwidth usage, so you can quickly identify those web applications that have been sending out bandwidth-intensive responses to users. For this, choose the **Bandwidth** option against this parameter.
Top N DD Limit	**This parameter applies only if the Show Top N in DD flag is set to 'Yes'.** By default, this parameter is set to 10, indicating that the detailed diagnostics will report the top-10 HTTP responses by default. You can change the 'N' in Top N by specifying any number of your choice in this text box.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the

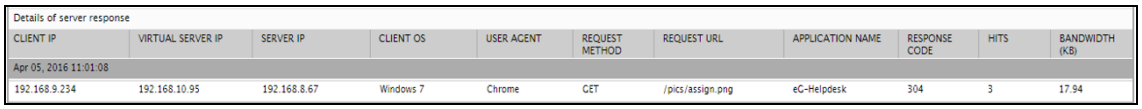
Parameter	Description
	detailed diagnosis capability for this test, you can do so by specifying *none* against **DD Frequency**.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of responses of this status sent by this NetScaler appliance.	Number	Compare the value of this measure across statuses to know which response is received often. Use the detailed diagnosis of this measure to know which application is sending out the maximum number of error responses and/or which application is sending out bandwidth-intensive responses.
Bandwidth	Indicates the total size of the responses of this status sent by this appliance.	KB	Compare the value of this measure across statuses to know which type of responses is consuming bandwidth excessively.
Avg render time	Indicates the elapsed time, from when the browser starts to receive the first byte of a response of this type until either all page content has been rendered	Msecs	Compare the value of this measure across response types to know which type of response is delaying page rendering.

Measurement	Description	Measurement Unit	Interpretation
	or the page load action has timed out.		

For a specific response type, the detailed diagnosis of the Hits measure reveals the web applications that sent out HTTP responses of that type, and the URLs for which the applications sent out such responses. The count of the responses of that type and average bandwidth usage of the responses is also reported, so you can quickly identify the web applications and request URLs that returned the maximum number of error responses and bandwidth-intensive responses. The details of the client to which such responses were sent will also be reported as part of detailed diagnostics.



Details of server response

CLIENT IP	VIRTUAL SERVER IP	SERVER IP	CLIENT OS	USER AGENT	REQUEST METHOD	REQUEST URL	APPLICATION NAME	RESPONSE CODE	HITS	BANDWIDTH (KB)
Apr 05, 2016 11:01:08										
192.168.9.234	192.168.10.95	192.168.8.67	Windows 7	Chrome	GET	/pics/assign.png	eG-Helpdesk	304	3	17.94

Figure 3.13: The detailed diagnosis of the Hits measure reported by the Citrix Web HTTP Response Status test

3.5.4 Citrix Web Operating Systems Test

When measuring user experience with web applications, it would also help administrators to know which client operating systems are widely used for accessing the applications and whether bandwidth usage and page rendering is impacted by the operating system in use. This way, administrators will be able to identify bandwidth-intensive operating systems and those that are inherently slow in rendering web pages. Users may then be discouraged to use such operating systems. The **Citrix Web Operating Systems** test enables administrators make such recommendations. This test auto-discovers the operating systems used by clients and reports the requests received, bandwidth consumed, and page rendering time per operating system. In the process, the test points to the most popular, the most bandwidth-intensive, and the slowest (in terms of page rendering) operating systems.

Target of the test : An AppFlow-enabled NetScaler Appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for every client operating system

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_DIR>\NetFlow\config directory.

Parameter	Description
	3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests received from clients running this operating system.	Number	Compare the value of this measure across operating systems to identify the most popular operating system.
Bandwidth	Indicates the total amount of data received from clients running this operating system.	KB	Compare the value of this measure across operating systems to know which operating system has been consistently consuming more bandwidth than the rest.
Avg render time	Indicates the elapsed time, from when clients running this operating system start to receive the first byte of a response until either all page content has been	Msecs	Compare the value of this measure across operating systems to know which operating system is taking too much time to render web pages.

Measurement	Description	Measurement Unit	Interpretation
	rendered or the page load action has timed out.		

3.5.5 Citrix Web User Agents Test

Users can connect to web applications using many client devices –web browsers, mobile phones, tablets, etc. Each such client device is called a user agent. Users using certain types of client devices/user agents may be engaged in bandwidth-intensive communication over the web, scarring the experience of other users to the web applications. Similarly, certain types of client devices - eg., some browser types like Firefox, Chrome etc. - may not be compatible with certain web applications. Users using such devices may therefore experience slowness when some web pages are rendered.

To assure users of the best experience with web applications at all times, administrators should track requests from the different client devices/user agents in use, measure the bandwidth usage and page rendering time of each device type, and share with users the list of client devices that may impact user experience with the web applications. The **Citrix Web User Agents** test helps administrators come up with such a list!

This test auto-discovers the types of client devices that are used for connecting to web applications, and reports the requests received from and bandwidth used by each device type. This way, the test leads administrators to those device types that are popular amidst users and those that are consuming bandwidth excessively. Additionally, the test also measures and reports the time taken by each client device type to render web pages, thus pointing to those device types that are much slower than the rest in page rendering. With the help of these metrics, administrators can arrive at a list of client devices that may be incompatible with their web applications.

Target of the test : An AppFlow-enabled NetScaler Appliance

Agent deploying the test : A remote agent

Outputs of the test : One set of results for each user agent/client device type that is connecting to the target NetScaler appliance

Configurable parameters for the test

Parameter	Description
Test period	How often should the test be executed. **It is recommended that you set the test period to 5 minutes. This is because, the eG AppFlow Collector is capable of capturing and aggregating AppFlow data related to the last 5 minutes only.**

Parameter	Description
Host	The host for which the test is to be configured.
Cluster IPs	**This parameter applies only if the NetScaler appliance being monitored is part of a NetScaler cluster.** In this case, configure this parameter with a comma-separated list of IP addresses of all other nodes in that cluster. If the monitored NetScaler appliance is down/unreachable, then the eG AppFlow Collector uses the **Cluster IPs** configuration to figure out which other node in the cluster it should connect to for pulling AppFlow statistics. Typically, the collector attempts to connect to every IP address that is configured against **Cluster IPs**, in the same sequence in which they are specified. Metrics are pulled from the first cluster node that the collector successfully establishes a connection with.
Enable Logs	This flag is set to **No** by default. This means that, by default, the eG agent does not create AppFlow logs. You can set this flag to **Yes** to enable AppFlow logging. If this is done, then the eG agent automatically writes the raw AppFlow records it reads from the collector into individual CSV files. These CSV files are stored in the *<EG_AGENT_INSTALL_DIR>\NetFlow\data\<IP_of_Monitored_NetScaler>\webappflow\actual_csv* folder on the eG agent host. These CSV files provide administrators with granular insights into the web appflows, thereby enabling effective troubleshooting. **Note:** By default, the eG agent creates a maximum of 10 CSV files in the **actual_csv** folder. Beyond this point, the older CSV files will be automatically deleted by the eG agent to accommodate new files with current data. Likewise, a single CSV file can by default contain a maximum of 99999 records only. If the records to be written exceed this default value, then the eG agent automatically creates another CSV file to write the data. If required, you can overwrite these default settings. For this, do the following: 1. Login to the eG agent host. 2. Edit the **Netflow.Properties** file in the <EG_AGENT_INSTALL_DIR>\NetFlow\config directory. 3. In the file, look for the parameter, **csv_file_retention_count**. 4. This is the parameter that governs the maximum number of CSV files that can be created in the **auto_csv** folder. By default, this parameter is set to 10. If you want to retain more number of CSV files at any given point in time, you can

Parameter	Description
	increase the value of this parameter. If you want to retain only a few CSV files, then decrease the value of this parameter. 5. Next, look for the parameter, **csv_max_flow_record_per_file**. 6. This is the parameter that governs the number of flow records that can be written to a single CSV. By default, this parameter is set to 99999. If you want a single file to accommodate more records, so that the creation of new CSVs is delayed, then increase the value of this parameter. On the other hand, if you want to reduce the capacity of a CSV file, so that new CSVs are quickly created, then decrease the value of this parameter. 7. Finally, save the file.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Hits	Indicates the number of requests received from this type of client.	Number	Compare the value of this measure across client types to identify that client that is used by a wide cross-section of application users.
Bandwidth	Indicates the total amount of data received from clients of this type.	KB	Compare the value of this measure across clients to know which type of client has been consistently consuming more bandwidth than the rest.
Avg render time	Indicates the elapsed time, from when the browser on this device starts to receive the first byte of a response until either all page content has been rendered or the page load action has timed out.	msecs	Compare the value of this measure across client devices to know which type of device is seeing the maximum page rendering time.

About eG Innovations

eG Innovations provides intelligent performance management solutions that automate and dramatically accelerate the discovery, diagnosis, and resolution of IT performance issues in on-premises, cloud and hybrid environments. Where traditional monitoring tools often fail to provide insight into the performance drivers of business services and user experience, eG Innovations provides total performance visibility across every layer and every tier of the IT infrastructure that supports the business service chain. From desktops to applications, from servers to network and storage, from virtualization to cloud, eG Innovations helps companies proactively discover, instantly diagnose, and rapidly resolve even the most challenging performance and user experience issues.

eG Innovations is dedicated to helping businesses across the globe transform IT service delivery into a competitive advantage and a center for productivity, growth and profit. Many of the world's largest businesses use eG Enterprise to enhance IT service performance, increase operational efficiency, ensure IT effectiveness and deliver on the ROI promise of transformational IT investments across physical, virtual and cloud environments.

To learn more visit www.eginnovations.com.

Contact Us

For support queries, email support@eginnovations.com.

To contact eG Innovations sales team, email sales@eginnovations.com.