Monitoring Apache Web Server

eG Innovations Product Documentation

www.eginnovations.com

Total Performance Visibility

Table of Contents

Table of Figures

Chapter 1: Introduction

Apache Web Server is a web server based on the Apache Software Foundation's Apache HTTP Server that runs on AIX, HP-UX, Linux, Solaris, Windows NT, and z/OS. The web servers are the heart of IT infrastructures in various domains - Healthcare, Banking, Trading, Logistics, etc. To ensure scalability and high performance, most web sites are being architected to use the multi-tier model - i.e., with the web server (IIS, Apache, etc.) functioning as the front-end, the middleware application server (J2EE, .Net based, etc.) that hosts the business logic functioning as the mid-tier, and a database server (SQL, Oracle, etc.) as the backend. In such architectures, the web server plays a pivotal role since all users to the other tiers are routed via the web server and hence, any slowdown or problem in the web server tier can adversely impact the end user experience.

The availability of a web site and the response time for user accesses to the site are the most critical metrics of web performance. Both these metrics may vary depending from one website to another and even from one transaction to another.

The eG adopts a unique two-pronged approach for web transaction monitoring on the Apache Web server. The external agent uses request emulation to assess the user experience from different locations. By doing so, the external agent captures the effect of the network latency and the server-side processing time on the end user experience.

To quantify the server processing times for real user requests (not emulated requests), the eG internal agent deploys a . This technology enhances web servers with the capability to track HTTP/HTTPS requests to a web server and the corresponding responses. For each transaction that is configured for monitoring, the web adapter analyzes the request URLs and responses to report various metrics relating to individual web transactions in real-time.

To ensure the uninterrupted operations of the Apache web server, administrator should continuously track overall health indicators such as the availability of the web server, the data processing ability of the server, the data traffic handled by the server, etc. To cater to the requirements of such administrators, eG Enterprise provides a specialized Apache Web server model. The statistics reported by this model, enable administrators to find out accurate answers to server's health related queries.

Chapter 2: How does eG Enterprise Monitor Apache Web Server?

eG Enterprise monitors the Apache web server in an agent-based manner using a specialized monitoring model. All that is required for this is a single eG agent deployed on the target web server. This eG agent pulls out the wealth of information about health of the web servers.

To measure the availability and overall health of the web server, the pre-requisites outlined in the topic below Section **2.1** should be fulfilled.

Configuring Apache Web Server for Monitoring by eG Enterprise

2.1 Configuring Apache Web Server for Monitoring by eG Enterprise

To pull out metrics related to the health of the Apache web server, the eG agent accesses a specific URL on the Apache server that contains the required metrics. To allow the eG agent to access this URL, you need to ensure that the following entries in the *<Apache_home>\confi\HTTPd.conf* file are uncommented (or enabled).

```
LoadModule status_module modules/mod_status.so
<IfModule mod_status.c>
ExtendedStatus On
</IfModule>
<Location /server-status>
SetHandler server-status
Order deny,allow
Deny from all
Allow from <domain name to give access>
</Location>
```

In case of the Apache web server v2.2 however, you will have to append the following entries to the *HTTPd.conf* file, soon after uncommenting the *LoadModule status_module modules/mod_status.so* entry:

```
<IfModule mod_status.c>
<Location /server-status>
SetHandler server-status
```

```
Order deny,allow
Deny from all
Allow from <domain name to give access>
</Location>
</IfModule>
```

While uncommenting or inserting (depending upon the version of Apache being monitored) the aforesaid block, make sure that the **<domain name to give access>** is configured with the fully qualified domain name that should be permitted to access the URL on the Apache web server. Alternatively, you can even specify the IP address of a particular host that should be granted access, or a space-separated list of 'allowed' IP addresses. Since it is the eG agent that should be allowed access to the URL, specify the fully qualified name of the domain to which the eG agent belongs and/or the IP address of the eG agent in **<domain name to give access>**. For example, your entry can read as follows:

```
Allow from mas.eginnovations.com
(OR)
Allow from mas.eginnovations.com 192.168.8.56
```

This will be the local host's IP/host name in case of an internal agent, or the IP/host name of a remote Windows host in case of a remote agent.

In case of an Apache web server v2.4, you need to uncomment (or enable) the following lines in the *<Apache_home>\confi\HTTPd.conf* file:

```
LoadModule status_module modules/mod_status.so
<IfModule mod_status.c>
<Location /server-status>
SetHandler server-status
Require ip <Host IP>
</Location>
</IfModule>
```

The **<Host IP>** in the entry highlighted in Red should be replaced with the IP address of the host that should be granted access, or a space-separated list of 'allowed' IP addresses. Since it is the eG agent that should be allowed access to the URL, specify the IP address of the eG agent here. This will be the local host's IP address in case of an internal agent, or the IP address of a remote Windows host in case of a remote agent. Finally, save the file.

2.1.1 Configuring eG Web Adapter for Apache Web Server on a 64-bit Linux Operating System

To configure the eG web adapter for an Apache web server on a 64-bit Linux host, do the following:

1. By default, the Java Runtime Environment (JRE) version 1.5 for 32-bit operating systems is bundled as part of the eG agent installable for Linux. To ensure that such an agent works smoothly on a 64-bit Linux operating system, follow the steps given below to change the JRE used by the eG agent, after deploying the standard Linux agent on the 64-bit Linux host:

   - Stop the eG agent.
   - Login to the eG agent host as the eG install user. You will currently be working in the **/opt/egurkha** directory.
   - Move the JRE directory that is used by the eG agent (by default) to another location, say **/tmp**.

     mv jre /tmp

2. If a 64-bit-compatible version of the JRE is already available on the eG agent host, provide a soft link to that directory using the following command:

   In -s <Full_path_to_the_directory_containing_the_64-bit-compatible_JRE> jre

   For instance, if the 64-bit version of JRE is available in the /opt/usr/JRE directory, then the command will be:

   In -s /opt/usr/JRE jre

3. On the other hand, if a compatible JRE does not pre-exist on the agent system, then download and install the 64-bit version of the JRE from the java.sun.com web site.
4. Then, provide a soft link to the JRE directory using the command indicated by step 4 above.
5. The eG agent for the 64-bit Linux host is bundled with the following shared libraries to be used by the eG web adapter, if configured on the host:

   - mod_eg24.so
   - mod_eg22.so
   - mod_eg2.so
   - libeg_reptr_cat.so

- libeg_reptr_total.so
- libeg_reptr_site.so

These files are available in the **/opt/egurkha/lib/lib64** directory on the host.

6. To enable the eG web adapter for Apache 2.0, following the steps given below:

- First, login to the Linux host as the Apache install user.
- Edit the *<APACHE_HOME>/conf/HTTPd.conf* file to append the following line:

```
LoadModule  eg2_module  modules/mod_eg2.so
```

- Copy the file *mod_ eg2.so* from the **/opt/egurkha/lib/lib64** directory to <APACHE_ HOME>/modules under <APACHE_INSTALL_USER>.
- Copy the *libeg*.so* files from the **/opt/egurkha/lib/lib64** directory to the **/opt/egurkha/lib d**irectory.
- Stop and restart the Apache server.

7. To enable the eG web adapter for Apache 2.2, following the steps given below:

- First, login to the Linux host as the Apache install user.
- Edit the *<APACHE_HOME>/conf/HTTPd.conf* file to append the following line:

```
LoadModule  eg2_module  modules/mod_eg22.so
```

- Copy the file *mod_ eg22.so* from the **/opt/egurkha/lib/lib64** directory to **<APACHE_ HOME>/modules** under **<APACHE_INSTALL_USER>**.
- Copy the *libeg*.so* files from the **/opt/egurkha/lib/lib64** directory to the **/opt/egurkha/lib** directory.
- Stop and restart the Apache server.

8. To enable the eG web adapter for Apache 2.4, following the steps given below:

- First, login to the Linux host as the Apache install user.
- Edit the *<APACHE_HOME>/conf/HTTPd.conf* file to append the following line:

```
LoadModule  eg2_module  modules/mod_eg24.so
```

- Copy the file *mod_ eg24.so* from the **/opt/egurkha/lib/lib64** directory to <APACHE_ HOME>/modules under <APACHE_INSTALL_USER>.

- Copy the *libeg*.so* files from the **/opt/egurkha/lib/lib64** directory to the **/opt/egurkha/lib directory**.
- Stop and restart the Apache server.

9. Start the eG agent.

2.1.2 Configuring Apache Web Server on Unix

eG Enterprise's unique web adapter technology enables individual transactions performed by users of a web site to be tracked in real-time without the need for explicit, expensive logging.

The web adapter must be configured for each and every web server that must be monitored by the eG agent. This adapter is part of the eG agent package for Solaris. In case of an Apache server on the other hand, a manual procedure needs to be followed in order to configure the web adapter.

To manually configure the eG web adapter on an Apache web server 1.x on Unix, do the following:

1. First, login to the Unix server as the Apache install user.
2. Edit the *<APACHE_HOME>/conf/HTTPd.conf* file to append the following lines to the end of the file:

```
LoadModule  eg1_module libexec/mod_eg1.so
```

3. Copy the *file mod_eg1.so* from the **/opt/egurkha/lib** directory to **<APACHE_HOME>/libexec** in the **<APACHE_INSTALL_USER>** directory.
4. Stop and restart the Apache server.

Note:

To configure the web adapter on Apache 1.x on HPUX/AIX servers, the procedure is almost the same as what has been discussed above; however, the only difference is that you will have to append the following lines to the end of the *<APACHE_HOME>/conf/HTTPd.conf* file:

```
LoadModule mod_egurkha libexec/mod_egurkha.so
```

To manually configure the eG web adapter on an Apache web server 2.0 on Unix, do the following:

1. First, login to the Unix server as the Apache install user.
2. Edit the *<APACHE_HOME>/conf/HTTPd.conf* file to append the following line:

```
LoadModule  eg2_module  modules/mod_eg2.so
```

3. Copy the file *mod_eg2.so* from the **/opt/egurkha/lib** directory to **<APACHE_HOME>/modules** under **<APACHE_INSTALL_USER>**.
4. Stop and restart the Apache server.

Note:

You cannot configure the web adapter on Apache web server 2.0 for HPUX.

To manually configure the eG web adapter on an Apache web server 2.2 on Unix, do the following:

1. First, login to the Unix server as the Apache install user.
2. Edit the *<APACHE_HOME>/conf/HTTPd.conf* file to append the following line:

```
LoadModule  eg2_module  modules/mod_eg22.so
```

3. Copy the file *mod_ eg22.so* from the **/opt/egurkha/lib** directory to **<APACHE_ HOME>/modules** under **<APACHE_INSTALL_USER>.**
4. Stop and restart the Apache server.

To configure the eG web adapter on Apache 2.2 on HPUX/AIX, follow the same procedure explained above.

To manually configure the eG web adapter on an Apache web server 2.4 on Linux, do the following:

1. First, login to the Unix server as the Apache install user.
2. Edit the *<APACHE_HOME>/conf/HTTPd.conf* file to append the following line:

```
LoadModule  eg2_module  modules/mod_eg24.so
```

3. Copy the file *mod_ eg24.so* from the **/opt/egurkha/lib** directory to **<APACHE_ HOME>/modules** under **<APACHE_INSTALL_USER>**.
4. Stop and restart the Apache server.

To configure the eG web adapter on Apache 2.4 on HPUX/AIX, follow the same procedure explained above.

2.1.3 Configuring Apache Web server on Windows

The eG web adapter can be configured on an Apache web server on Windows, using a manual configuration process only. The same has been discussed below.

To manually configure the eG web adapter on an Apache web server 1.x on Windows, do the following:

- First, login to the Windows server.
- Edit the *<APACHE_HOME>\conf\HTTPd.conf* file to append the following lines:

```
AddModule  mod_egurkha.c
```

```
LoadModule mod_egurkha  modules/mod_egurkha.dll
```

- Copy the file *mod_ egurkha.dll* from the <EG_ AGENT_ INSTALL_ DIR>\lib directory to <APACHE_HOME>\modules.
- Stop and restart the Apache server.

To manually configure the eG web adapter on an Apache web server 2.0 on Windows, do the following:

- First, login to the Windows server.
- Edit the *<APACHE_HOME>\conf\HTTPd.conf* file to append the following line:

```
LoadModule  egurkha_module  modules/mod_egurkha2_0.dll
```

- Copy the file mod_egurkha2_0.dll from the **<EG_AGENT_INSTALL_DIR>\lib** directory to **<APACHE_HOME>\modules**.
- Stop and restart the Apache server.

To manually configure the eG web adapter on an Apache web server 2.2 on Windows, do the following:

- First, login to the Windows server.
- Edit the *<APACHE_HOME>\conf\HTTPd.conf* file to append the following line:

```
LoadModule  egurkha_module  modules/mod_egurkha2_2.dll
```

- Copy the file *mod_egurkha2_ 2.dll* from the <EG_AGENT_INSTALL_DIR>\lib directory to <APACHE_HOME>\modules.
- Stop and restart the Apache server.

2.1.4 Customizing Applications for Monitoring by eG Enterprise's Web Adapter

The HTTP protocol specification provides various status codes that are used by web applications to indicate error conditions. For example, a response code of 404 indicates that a specific page was not found on the server. Likewise, a response code of 500 indicates a server-side processing error. As the Web has evolved to support a variety of complex applications that involve dynamic rather than static content, application developers have resorted to newer methods of informing users of application-level problems. For example, rather than returning a 404 response code to indicate missing content (which results in the browser throwing an error message), an application developer may choose to report the error by formatting it within a HTML page (i.e., by providing a 200 response code) and providing additional user-friendly error messages such as the email address of the web site's administrator. The side effect of this approach is that the large number of existing monitoring tools that primarily use the HTTP response code to detect application failures will not be able to effectively detect and report problem conditions.

To enable eG agents to detect and report on such application-specific problems, eG Enterprise's web adapter allows applications to use a specific HTTP header variable called Eg-Status to report error conditions to it. To report a specific problem, application developers should assign corresponding error codes (following the HTTP protocol specification's response code convention). For example, to report an application-specific error, an application that uses Java technologies (JSP, Servlets, etc.) can incorporate the following code to set the Eg-Status header value while generating a HTTP response:

```
response.setHeader("Eg-Status", "500");
```

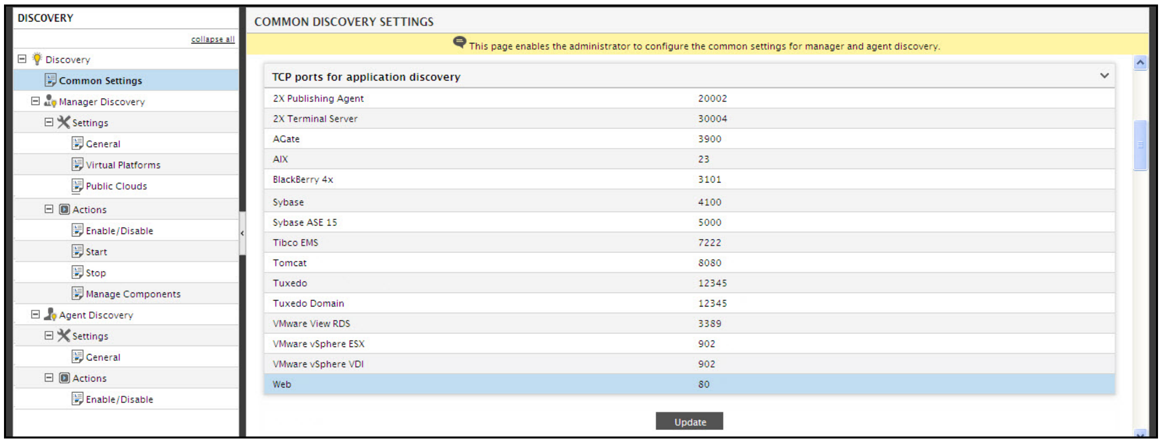
The eG web adapter searches the HTTP header of all responses generated by a web server. If the Eg-Status header exists, the value of this variable is used to override the HTTP response code value. This method allows application developers to indicate potential error conditions to the eG agents, without affecting the output being provided by their applications to users.

2.2 Managing the Apache Web Server Monitored using Web Adapter

eG Enterprise can automatically discover the Apache web server in your environment. To discover a web server automatically, the port on which it is running should be configured. To achieve this, do the following:

1. Log into the eG administrative interface as admin.
2. Following the menu sequence: Infrastructure-> Components -> Discovery will lead you to the **DISCOVERY** page.
3. Clicking on the **Common Discovery settings** in the left panel will lead you to the **COMMON DISCOVERY SETTINGS** page. Expanding the **TCP ports for application discovery** section

will help you to configure the port on which the web server is listening (See Figure 2.1). Once done, click on the **Update** button.



COMMON DISCOVERY SETTINGS

This page enables the administrator to configure the common settings for manager and agent discovery.

TCP ports for application discovery

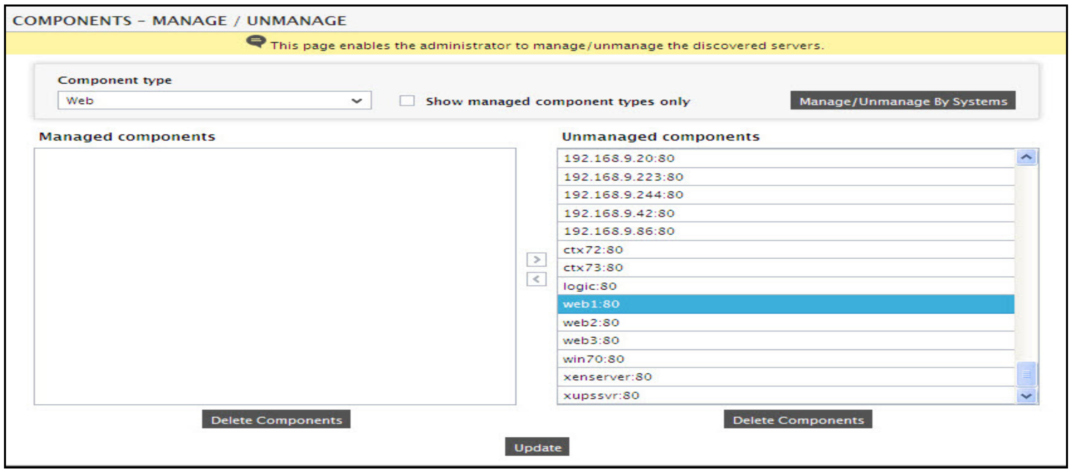
2X Publishing Agent	20002
2X Terminal Server	30004
AGate	3900
AIX	23
BlackBerry 4x	3101
Sybase	4100
Sybase ASE 15	5000
Tibco EMS	7222
Tomcat	8080
Tuxedo	12345
Tuxedo Domain	12345
VMware View RDS	3389
VMware vSphere ESX	902
VMware vSphere VDI	902
Web	80

Figure 2.1: Configuring the web server port for automatic discovery

4. After updating the changes, begin the discovery process by clicking on the **Start** option under the Actions node in the **DISCOVERY** tree.

5. Discovered components will then have to be managed manually using the **COMPONENTS - MANAGE/UNMANAGE** page that comes up on following the menu sequence: Infrastructure-> Components -> Manage/Unmanage.

6. In this screen, select **Web** server from the **Component type** drop down list as depicted by Figure 2.2 below.



COMPONENTS – MANAGE / UNMANAGE

This page enables the administrator to manage/unmanage the discovered servers.

Component type: Web

Show managed component types only

Managed components

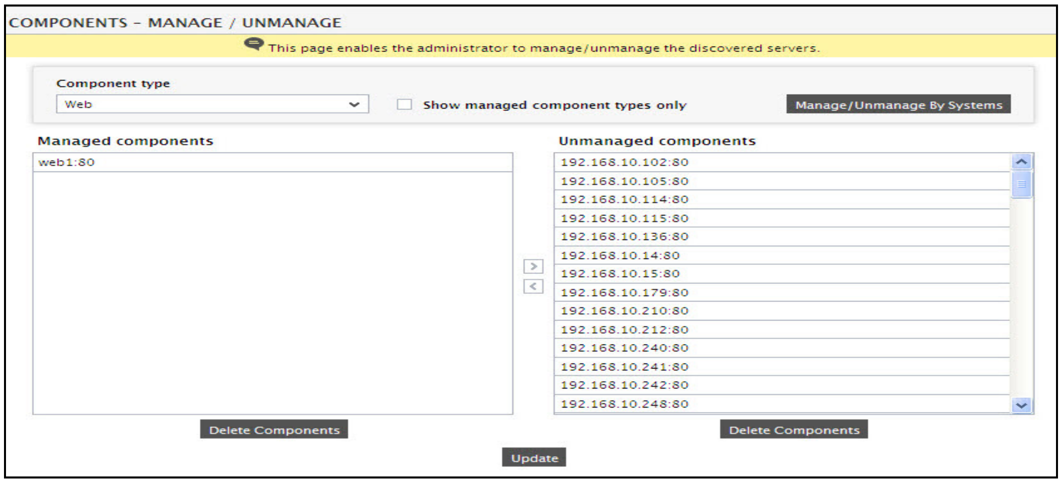
Unmanaged components

192.168.9.20:80
192.168.9.223:80
192.168.9.244:80
192.168.9.42:80
192.168.9.86:80
ctx72:80
ctx73:80
logic:80
web1:80
web2:80
web3:80
win70:80
xenserver:80
xupssvr:80

Figure 2.2: Selecting Web server from the drop-down list

7. The host names / IP addresses of the discovered but unmanaged web servers will then be

populated in the **UNMANAGED COMPONENTS** list. To manage the web server, select it from the list, click the << **Manage** button and add it to the **MANAGED COMPONENTS** list as shown in Figure 2.3 below.


COMPONENTS - MANAGE / UNMANAGE

This page enables the administrator to manage/unmanage the discovered servers.

Component type: Web

Show managed component types only

Managed components:
web1:80

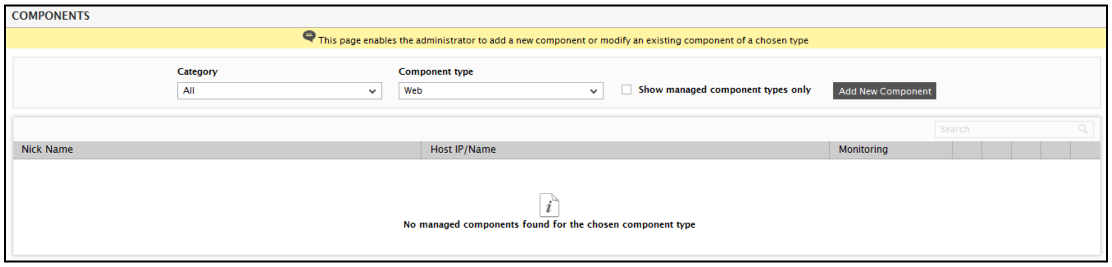
Unmanaged components:
192.168.10.102:80
192.168.10.105:80
192.168.10.114:80
192.168.10.115:80
192.168.10.136:80
192.168.10.14:80
192.168.10.15:80
192.168.10.179:80
192.168.10.210:80
192.168.10.212:80
192.168.10.240:80
192.168.10.241:80
192.168.10.242:80
192.168.10.248:80

Figure 2.3: Managing the Apache web server manually

8. Finally, register the changes by clicking the **Update** button.

9. If the web server is not automatically discovered, manually add it to the environment using the **COMPONENTS** page. Components so added will automatically find a place in the **MANAGED COMPONENTS** list in the **COMPONENTS - MANAGE/UNMANAGE** page above (see Figure 2.3). To access the **COMPONENTS** page, follow the menu sequence: Infrastructure -> Components -> Add/Modify.

10. Here, select **Web** server from the **Component type** drop-down list (see Figure 2.4) and then, click the **Add New Component** button to add a new Apache Web server.


COMPONENTS

This page enables the administrator to add a new component or modify an existing component of a chosen type

Category: All

Component type: Web

Show managed component types only

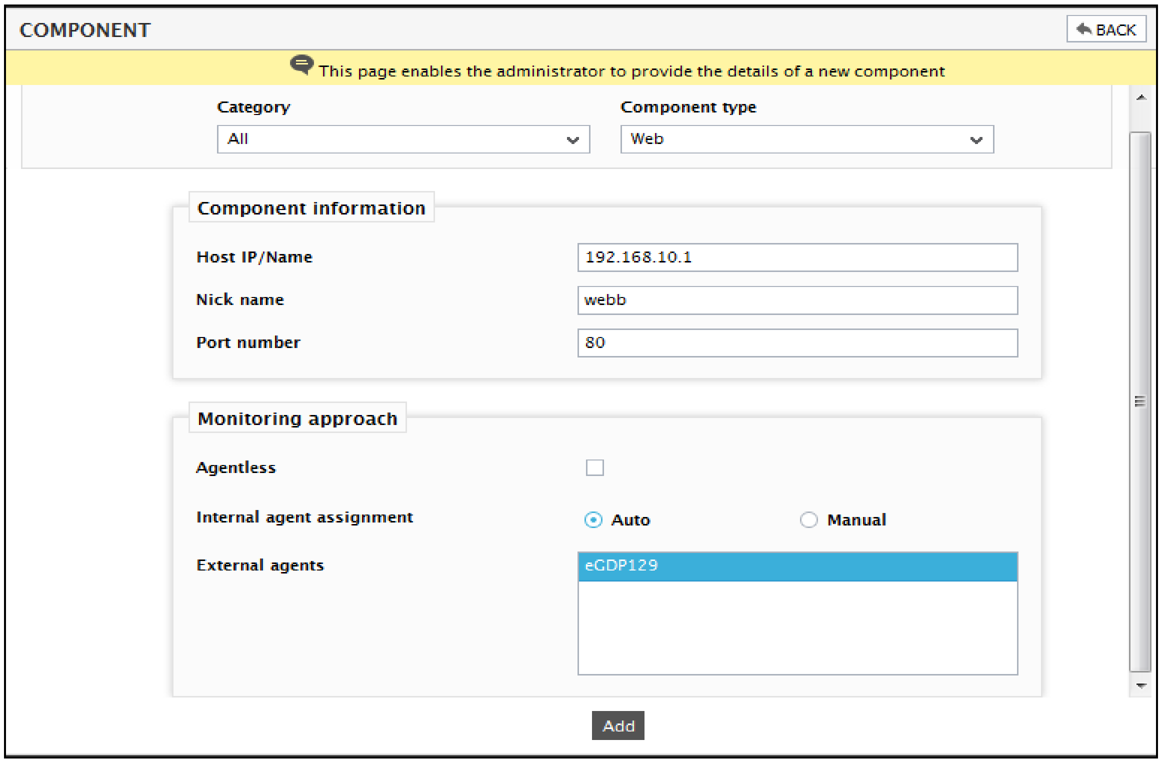
Nick Name	Host IP/Name	Monitoring

No managed components found for the chosen component type

Figure 2.4: Selecting the Web server option from the drop-down list in the ADD/MODIFY page

11. In the **COMPONENT** page that appears, provide the details requested as depicted by Figure 2.5. If a valid hostname is specified in Figure 2.5, make sure that this name can be resolved to the corresponding IP address via the DNS server of the target infrastructure. Also, indicate whether

the component being added is to be monitored in an agentless manner or not. This option will be available to you only if the eG license enables the agentless monitoring capability. Moreover, if the **Agentless** option in Figure 2.5 is set to **No**, then an **Internal agent assignment** option will appear. If this is set to **Manual**, then you can associate multiple IPs/nicknames on a host with a single internal agent. The default selection **Auto** indicates that every IP/nick name on a host will be associated with a separate agent. For more details about the **Agentless** and Internal agent assignment options, refer to the *Administering eG Enterprise Suite manual*. Finally, select the **External agent** that will monitor the component being added from an external perspective, and click the **Add** button in Figure 2.5 to register the changes.



This page enables the administrator to provide the details of a new component

Category: All
Component type: Web

Component information

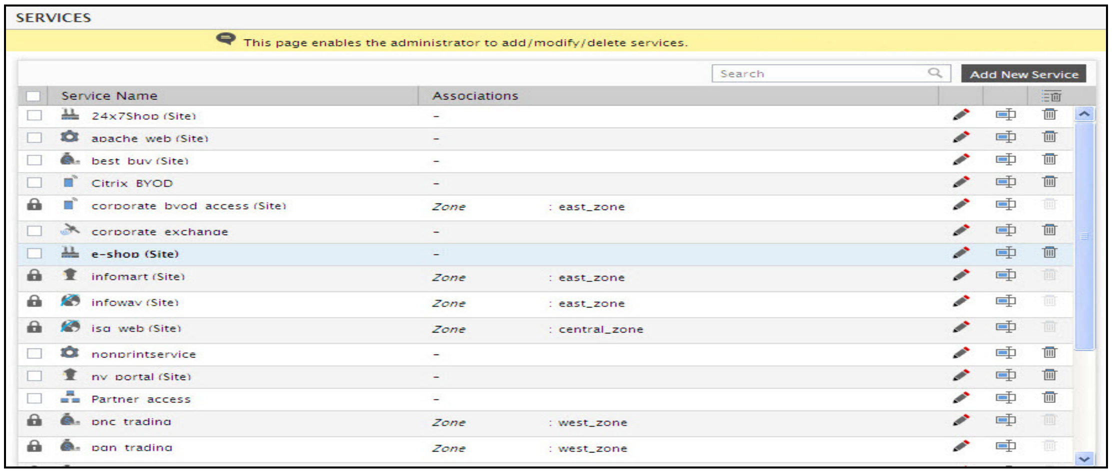
Host IP/Name	192.168.10.1
Nick name	webb
Port number	80

Monitoring approach

Agentless	☐
Internal agent assignment	[✓ Auto] Manual
External agents	eGDP129

Figure 2.5: Providing the new server details

12. Then, proceed to configure the services for the web server. A service can be a group of independent components / components that belong to a segment topology / a web site that can be hosted on one or more web servers. In other words, a web site is a subset of a service. The various services that users can avail via a web site are referred to as **transactions**. To configure services, first, follow the menu sequence, Infrastructure ->Services -> Configuration. This will open the **SERVICES** page (see Figure 2.6).



SERVICES

This page enables the administrator to add/modify/delete services.

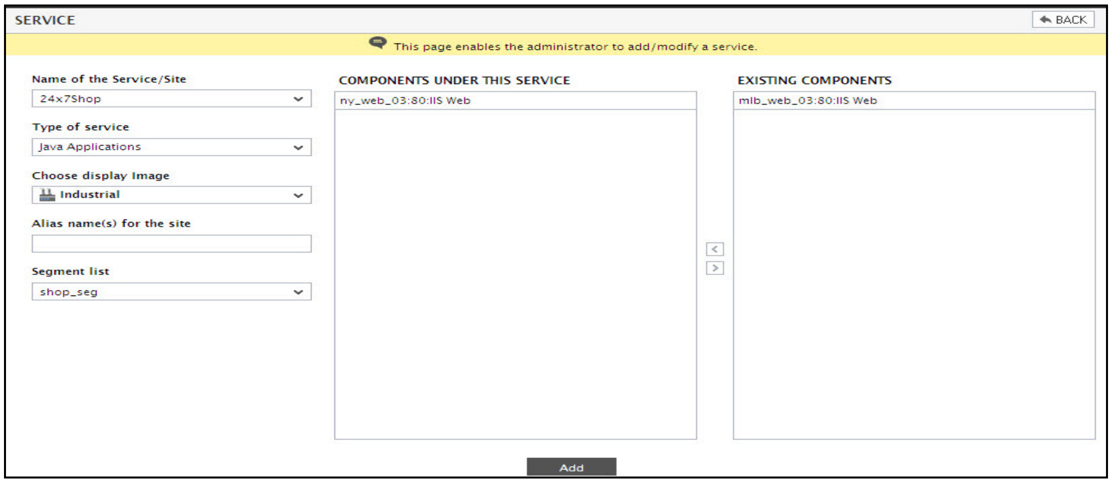
Service Name	Associations	
24x7Shop (Site)	-	
apache web (Site)	-	
best buy (Site)	-	
Citrix BYOD	-	
corporate byod access (Site)	Zone	: east_zone
corporate exchange	-	
e-shop (Site)	-	
infomart (Site)	Zone	: east_zone
infoway (Site)	Zone	: east_zone
isa web (Site)	Zone	: central_zone
nonprintservice	-	
ny portal (Site)	-	
Partner access	-	
pnc trading	Zone	: west_zone
ppn trading	Zone	: west_zone

Figure 2.6: A page listing existing web sites

13. Then, click on the **Add New Service** button on the screen. In the screen that appears (see Figure 2.7), specify thename of the service/site in the **Name of the Service/Site** text box. Then, from the **Is this service a website** list box, select **Yes**, so as to indicate that the new service is a web site.

    Note:

    While adding a service that is not a web site, the **No** option needs to be selected from the **Is this service a website** list box.



SERVICE

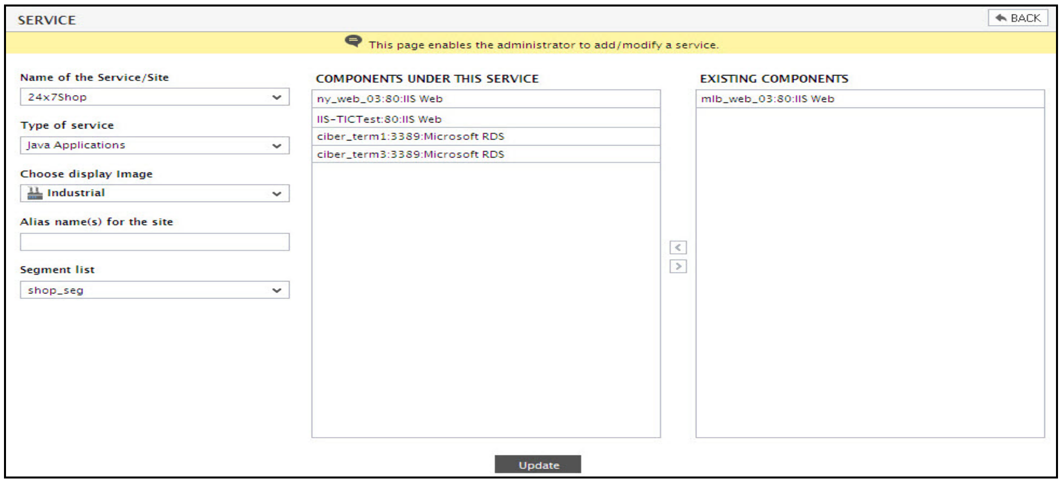
This page enables the administrator to add/modify a service.

Name of the Service/Site	24x7Shop
Type of service	Java Applications
Choose display Image	Industrial
Alias name(s) for the site	
Segment list	shop_seg

COMPONENTS UNDER THIS SERVICE	EXISTING COMPONENTS
ny_web_03:80:IIS Web	mlb_web_03:80:IIS Web

Figure 2.7: Adding a new service

14. Finally, click the **Update** button. Upon clicking, Figure 2.8 will appear, which will help an administrator configure a service that is a web site.


SERVICE
BACK
This page enables the administrator to add/modify a service.
Update

Name of the Service/Site: 24x7Shop
Type of service: Java Applications
Choose display Image: Industrial
Alias name(s) for the site:
Segment list: shop_seg

COMPONENTS UNDER THIS SERVICE	EXISTING COMPONENTS
ny_web_03:80:IIS Web	mlb_web_03:80:IIS Web
IIS-TICTest:80:IIS Web	
ciber_term1:3389:Microsoft RDS	
ciber_term3:3389:Microsoft RDS	

Figure 2.8: Configuring a web site

15. A single site can be addressed by various other names in the environment (e.g., www.abc.com may also be accessed as www.abc.com:80, abc.com, us.abc.com, 172.169.10.20 etc.). These names (or IP address:port combinations) can be specified in the **Alias name(s) for the site** text box. To ensure that all requests to a website are captured, it is essential to ensure that all the alias names for a site are specified accurately. The administrators can specify a maximum of six alias names, each of which should be comma separated. While multiple alias names can be specified for a site, in the monitor interface, all the statistics pertaining to this web site will be reported using its site name displayed in the **Name of the site** text box.

    Note:

    Alias names are applicable to web sites alone. Therefore, while configuring a service that is *not* a web site, the **Alias name(s) for the site** text box will not appear.

16. The **Segment list** box contains the list of fully configured segments in the target environment that contain atleast a single web or application server. The site can be associated with any of these segments. The **Independent components** option in this list box enables the administrator to associate a site to a single web or application server that does not form a part of the segment topology. By default, a managed web server will be treated as an independent web server by the eG Enterprise system. Therefore, select the **Independent components** option from the **Segment list**.

    Note:

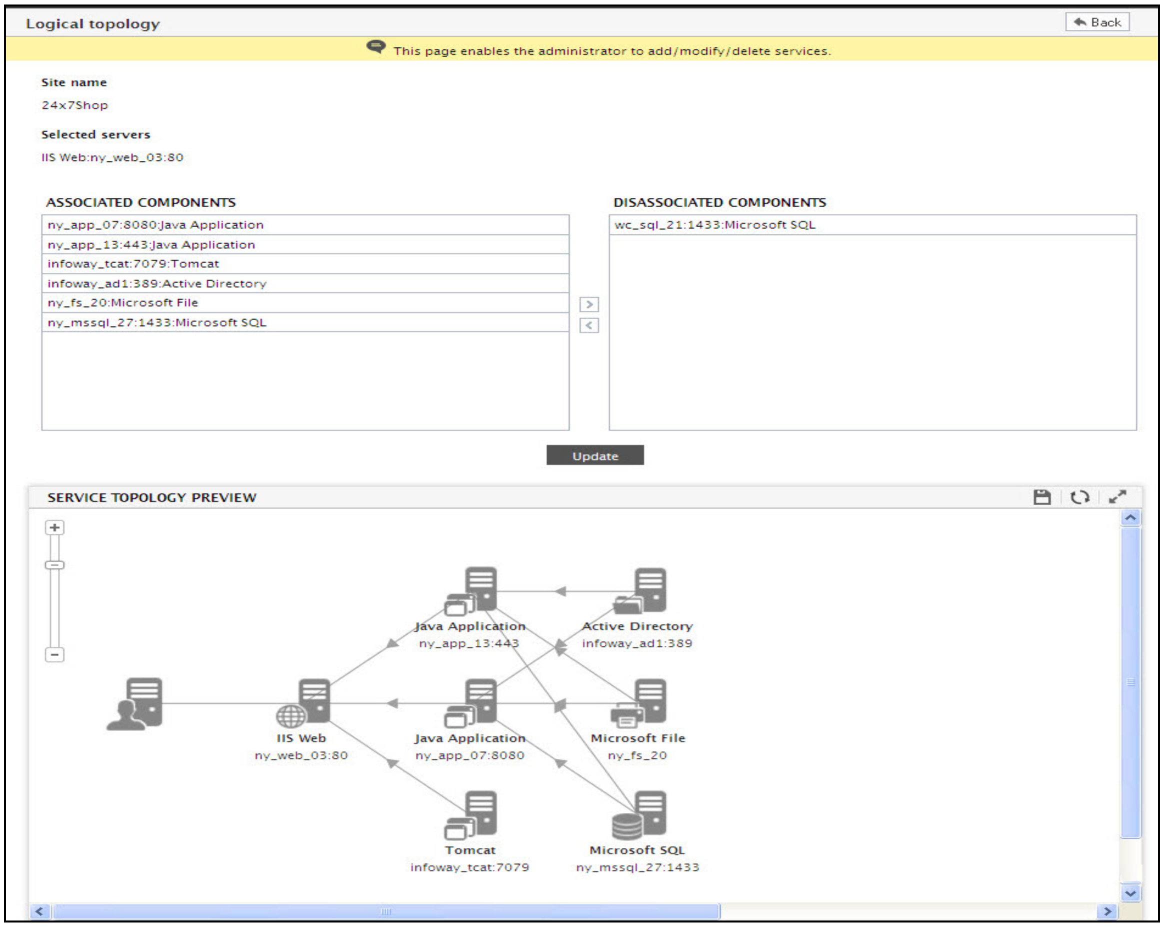
    In case of services that are *not* web sites, the **Segment list** box will list all the fully configured segments in the target environment – not just the segments that contain web / application servers.

17. Once a site is associated with a segment, the user interface lists the web and application servers that form a part of the selected segment (or, if the **Independent_servers** option is chosen, the list of independent web and application servers in the target environment), in the **EXISTING COMPONENTS** list box. The components associated with the site are available in the **COMPONENTS UNDER NEW SITE** list box. The administrator can associate an existing component with the site by selecting the component and then clicking the **Add >>** button. Similarly, an associated component can be removed by clicking the **<< Remove** button.

    Note:

    Unlike web sites that can be associated only with web / application servers, services that are **not** web sites can be associated with any component in the selected segment, or any independent component.

18. Finally, click the **Update** button to register the changes. On doing so, the following screen will pop up:


Logical topology

This page enables the administrator to add/modify/delete services.

Site name
24x7Shop

Selected servers
IIS Web:ny_web_03:80

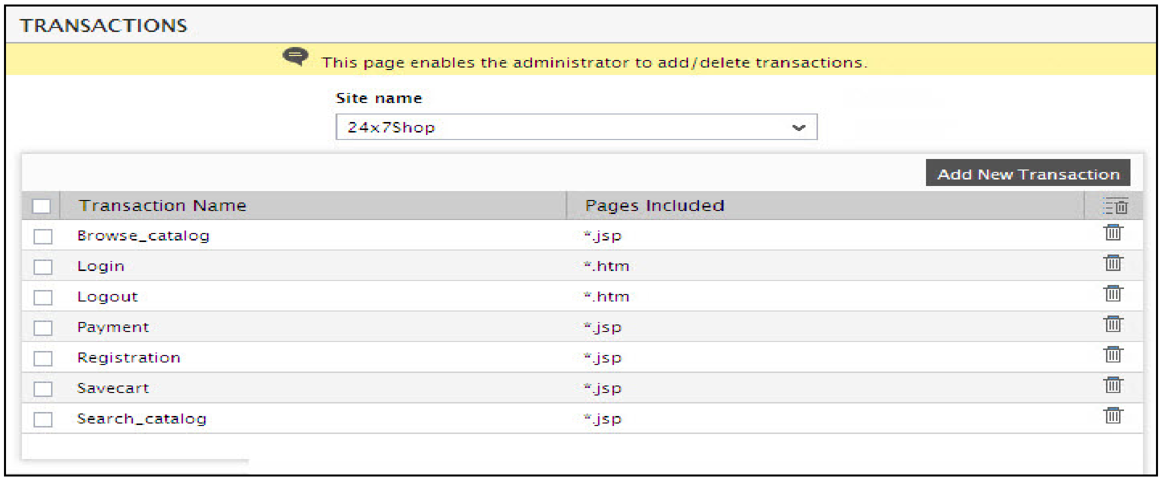
ASSOCIATED COMPONENTS
ny_app_07:8080:Java Application
ny_app_13:443:Java Application
infoway_tcat:7079:Tomcat
infoway_ad1:389:Active Directory
ny_fs_20:Microsoft File
ny_mssql_27:1433:Microsoft SQL

DISASSOCIATED COMPONENTS
wc_sql_21:1433:Microsoft SQL

Figure 2.9: Configuring the dependencies of a web site

19. A web site inherits all of the dependencies of the web server(s) with which it is associated. In many instances, a web site may not use all the components that a web server on which it is hosted, is configured to use. To support such a scenario, the eG Enterprise suite allows the administrator to disassociate specific components for a web site. Figure 2.9 indicates the configuration of dependencies for a web site. The components associated with a site are shown in the **ASSOCIATED COMPONENTS** box and the non-related ones in the **EXCLUDED COMPONENTS** box. When a site is added for the first time, all the components that are associated with the corresponding web server (s) are available in the **ASSOCIATED COMPONENTS** list box. The administrator can associate or disassociate the components using the **Associate** or the **Disassociate** buttons. In our example, however, the web server associated with the site is an independent one. Hence, simply click the **Update** button and proceed to configure transactions.

20. Administrators can configure transactions to reflect the key operations performed by users of the web site. For performing this activity, first, open the **LIST OF SITES AND TRANSACTIONS** page (see Figure 2.10) using the following menu sequence: Infrastructure- >Service- >Transactions.


TRANSACTIONS
This page enables the administrator to add/delete transactions.
Site name
24x7Shop
Add New Transaction

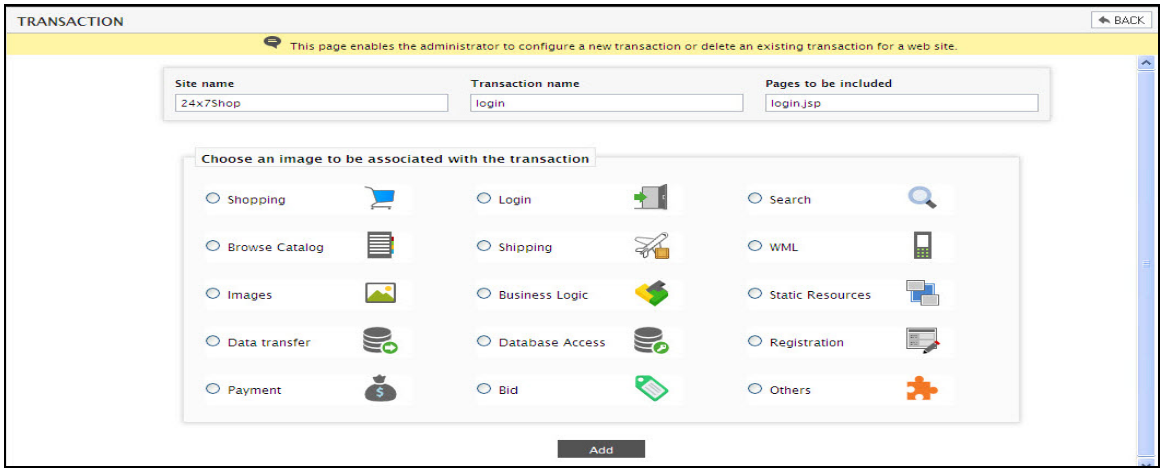
Transaction Name	Pages Included
Browse_catalog	*.jsp
Login	*.htm
Logout	*.htm
Payment	*.jsp
Registration	*.jsp
Savecart	*.jsp
Search_catalog	*.jsp

Figure 2.10: A page listing existing transactions of a web site

Note:

Transactions can be configured for web sites only. Therefore, Figure 2.11 will not list the services that are not websites.

21. Then, click the **Add/Delete Transaction** button therein, thereby opening the following page:



This page enables the administrator to configure a new transaction or delete an existing transaction for a web site.

Site name	Transaction name	Pages to be included
24x7Shop	login	login.jsp

Choose an image to be associated with the transaction

○ Shopping	○ Login	○ Search
○ Browse Catalog	○ Shipping	○ WML
○ Images	○ Business Logic	○ Static Resources
○ Data transfer	○ Database Access	○ Registration
○ Payment	○ Bid	○ Others

Figure 2.11: Figure 2.11: Configuring a new transaction

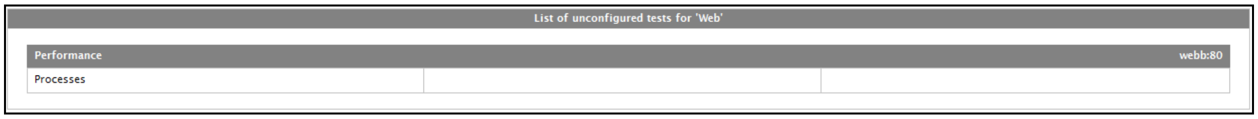
22. Here, specify the **Transaction Name** and the **Pages to be Included** (these are one or more regular expression patterns, where each pattern refers to a set of pages that constitute the transaction).

    For example: The transactions of a retail web site could be: login, registration, browsing of the product catalog, searching the catalog, adding to shopping cart, deleting items from the cart, payment, etc. The **PAGES TO BE INCLUDED** for the **Login** transaction could be represented by ***/jsps/Loginform.jsp**.

    Note:

    While mentioning the **PAGES TO BE INCLUDED**, ensure the following:

23. The page names should be prefixed by an * (asterisk) or a slash (/). If not, no measurements will be gathered from such pages.
24. You can also associate an image with a transaction, by choosing the same from the list below.
25. Then, to add the transaction, click on the **Add** button. Clicking on **Add** will take you back to the **NEW TRANSACTION DETAILS** page above, where you would be prompted to add another transaction.
26. Now, if you attempt to sign out, the **Processes** test information will flash on the screen, prompting you to configure the test.



List of unconfigured tests for 'Web'

Performance		webb:80
Processes		

Figure 2.12: Viewing the tests configuration table displaying the Processes test information

27. Clicking on the **Processes** test to configure it. To know how to configure the test, refer to *Monitoring Unix and Windows Servers* document.

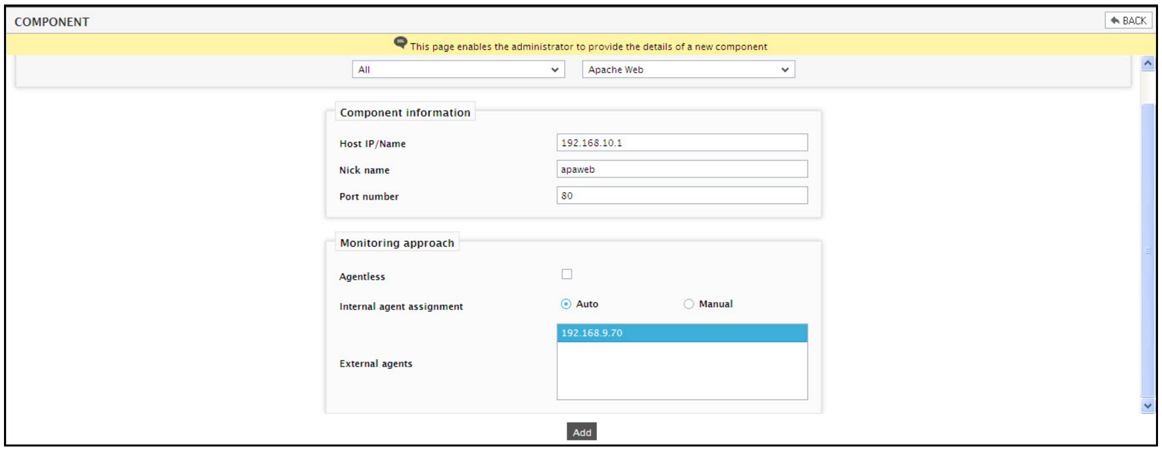
28. Finally, signout of the eG administrative interface.

2.3 Managing the Apache Web Server

To achieve this, do the following:

1. Login to the administrative interface of eG as an administrator (admin).

2. If the Apache web server is automatically discovered, then use the **COMPONENTS –MANAGE / UNMANAGE** page to manage the server. On the other hand, if the Apache web server is not discovered automatically, then either run discovery to get them discovered (Infrastructure -> Components -> Discover) or add them using the **COMPONENTS** page (Infrastructure -> Components -> Add/Modify) (see Figure 2.13). Components manually added will be automatically managed by the eG Enterprise system (see Figure 2.13).



This page enables the administrator to provide the details of a new component

All | Apache Web

Component information

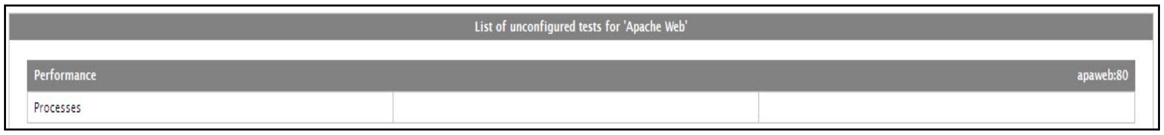
Host IP/Name	192.168.10.1
Nick name	apaweb
Port number	80

Monitoring approach

Agentless	☐
Internal agent assignment	[✓ Auto] Manual
External agents	192.168.9.70

Figure 2.13: Adding the details of a new Apache web server

3. Now, try to sign out of the user interface. Doing so, will bring up a page, which prompts you to configure the tests for the Apache web server.


List of unconfigured tests for 'Apache Web'

Performance		apaweb:80
Processes		

Figure 2.14: The list of unconfigured tests for the Apache web server

4. Click on the **Processes** test in Figure 2.14 to configure it. To know how to configure this test, refer to *Monitoring Unix and Windows Servers* document.
5. Finally signout of the eG administrative interface.

Chapter 3: Monitoring Apache Web Server

Using the Apache web server (see Figure 3.1) monitoring model, administrators can keep an eye on the overall health indicators such as the availability of the Apache web server, the data processing ability of the server, the data traffic handled by the server, etc., monitored. .

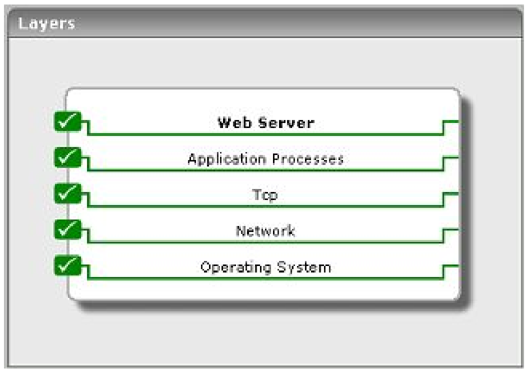


Figure 3.1: Layer model of the Apache web server

Each of the layers depicted by Figure 3.1 is associated with a series of tests that report on key Apache-specific performance parameters, and answers the following performance-related queries:

- Is the web server available?
- How frequently is the web server accessed?
- Is CPU utilization of the web server optimal?
- How is the traffic on the web server?
- How well does the web server process requests? Are there any bottlenecks in processing?

As the **Tcp**, **Network**, and **Operating System** layers have been discussed in great detail in the *Monitoring Unix and Windows Servers* documentmonitoring model, the sections to come will discuss the **Application Processes** and **Web Server** layers alone.

3.1 The Application Processes Layer

This layer depicts the states of the different processes that must be executing for the Web server to be available.

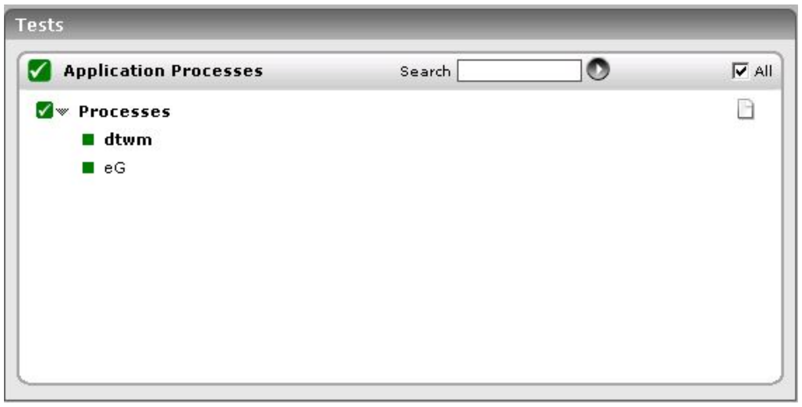


Figure 3.2: The tests mapped to the Application Processes layer

In addition, the layer can also be optionally configured to monitor the validity of the SSL certificates (if any) that may have been installed on the Web server. For this, an **SSL Certificate** test will have to be enabled. The details of this test have been provided below.

3.1.1 SSL Certificate Test

This test reports how long (in days) the SSL certificates that have been configured for monitoring will remain valid.

This test is disabled by default. To enable the test, go to the **ENABLE / DISABLE TESTS** page using the menu sequence : Agents -> Tests -> Enable/Disable, pick the desired **Component type**, set *Performance* as the **Test type**, choose the test from the **DISABLED TESTS** list, and click on the **<** button to move the test to the **ENABLED TESTS** list. Finally, click the **Update** button.

Target of the test : An Apache web server

Agent deploying this test : An internal agent

Outputs of the test : One set of outputs for every Target and/or every Targetfile and/or the unique key assigned to each certificate in the specified Keystore File.

Configurable parameters for the test

Parameters	Description
Test period	This indicates how often should the test be executed.
Host	The host for which the test is to be configured.
Targets	If you want to monitor specific SSL-enabled web sites, then, provide a comma-separated list of *{HostIP/Name}:{Port)* pairs, which represent the web sites to be

Parameters	Description
	monitored. For example, *192.168.10.7:443,192.168.10.8:443*. The test connects to each IP/port pair and checks for the validity of the certificate associated with that target. One set of metrics is reported for each target. The descriptor represents the common name (CN) value of the SSL certificate. By default, this parameter is set to the *<IP_of_the_monitored_web/application_server>:<Port_on_which_the_server_listens>*. If you do not want to monitor the validity of certificates based on configured targets, set this parameter to *none*.
Targetfiles	To monitor specific certificate files, provide a comma-separated list of file paths for the SSL certificates that are to be monitored in the Targetfiles text box. For example, *C:\server.crt, D:\admin.crt*. The test reads the SSL Certificates for the web sites that are to be monitored from this location and checks for the validity. If you do not want to check the validity of specific certificate files, set this parameter to *none*.
Keystore File	A keystore is a database (usually a file) that can contain trusted certificates and combinations of private keys with their corresponding certificates. If you are looking to monitor the certificates contained within a keystore file, then provide the full path to this file in the Keystore File text box. For example, the location of this file may be: *C:\egurkha\manager\tomcat\webapps\eGmanager.bin*. In this case, the test automatically accesses each of the certificates that the specified keystore contains, and checks its validity. If you do not want to monitor the certificates in a keystore, set this parameter to *none*.
Keystore Password	If a Keystore File is provided, then, in this text box, provide the password that is used to obtain the associated certificate details from the Keystore File. If *none* is specified against Keystore File, then, enter *none* here as well.
Confirm Password	Confirm the Keystore Password by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
SSL certificate validity	Indicates the number of days from the current day for which this SSL certificate will be valid.	Number	

3.2 The Web Server Layer

The tests associated with the **Web Server** layer monitor the availability of the Apache web server and measures the traffic on the Apache web server, its current CPU load, how well the Apache server handles requests, etc (see Figure 3.3).

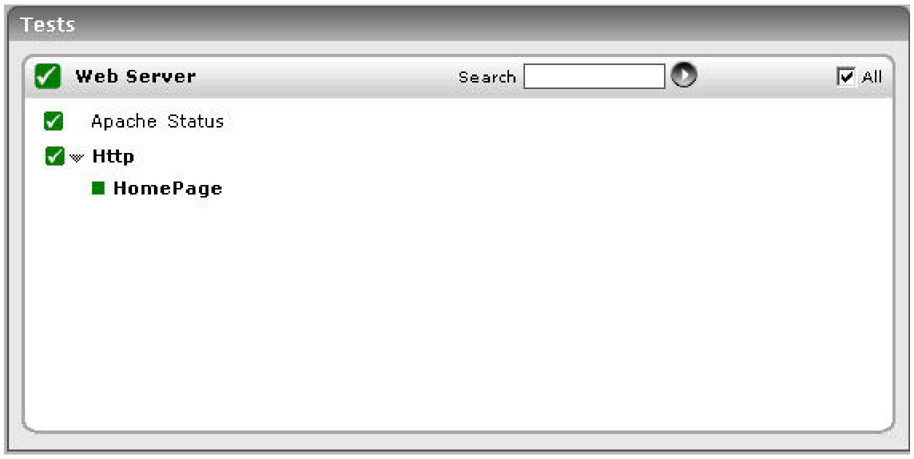


Figure 3.3: The tests associated with the Web Server layer

3.2.1 Apache Status Test

The Apache Status test reveals critical performance statistics pertaining to an Apache web server. This test, upon execution, accesses a specific URL on the Apache server, which contains the required metrics.

Target of the test : An Apache web server

Agent deploying this test : An internal agent

Outputs of the test : One set of results for the Apache web server being monitored

Configurable parameters for the test

Parameters	Description
Test period	This indicates how often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port to which the specified host listens
URL	In the this text box, the URL to be accessed by this test for extracting the Apache web server's performance statistics, will be displayed by default. The URL is: *http://{Apache web server host}:{Apache web server port}/server-status*.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Access rate	Indicates the rate at which the users access the web server.	Accesses/Sec	
Data traffic	Indicates the rate at which the web server transmits data.	KB/Sec	
CPU load	Indicates the percentage of CPU used by the web server.	Percent	If the value of this measure increases with time, it could be a cause for concern.
Current requests	Indicates the number of requests that are currently being processed by the web server.	Number	
Idle workers	Indicates number of idle worker threads on the web server, currently.	Number	A very high value of this measure could indicate a processing bottleneck.

3.2.2 Apache Server Status Test

This test reveals critical performance and uptime statistics pertaining to an Apache web server. This test, upon execution, accesses a specific URL on the Apache server, which contains the required metrics. Using this test, administrators can figure out how well the processes are being procesed in the server, how many processes are currently idle, the processes that are in *KEEPALIVE* and Logging state etc. This way, this test helps administrators to identify processing bottlenecks, if any.

Target of the test : An Apache web server

Agent deploying this test : An internal agent

Outputs of the test : One set of results for the Apache web server being monitored

Configurable parameters for the test

Parameters	Description
Test period	This indicates how often should the test be executed.

Parameters	Description
Host	The host for which the test is to be configured.
Port	The port to which the specified host listens
URL	In the this text box, the URL to be accessed by this test for extracting the Apache web server's performance statistics, will be displayed by default. The URL is: *http://{Apache web server host}:{Apache web server port}/server-status*.
system_property_key	
system_property_value	
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
Total uptime of Apache server	Indicates the total number of days that the server has been up since its last reboot.	Days	Administrators may wish to be alerted if a server has been running without a reboot for a very long period. Setting a threshold for this measure allows administrators to determine such conditions.
Uptime during the last measure period	Indicates the time period that the server has been up since the last time this test ran.	Secs	If the server has not been rebooted during the last measurement period and the agent has been running continuously, this value will be equal to

Measurement	Description	Measurement Unit	Interpretation
			the measurement period. If the server was rebooted during the last measurement period, this value will be less than the measurement period of the test. For example, if the measurement period is 300 secs, and if the server was rebooted 120 secs back, this metric will report a value of 120 seconds. The accuracy of this metric is dependent on the measurement period - the smaller the measurement period, greater the accuracy.
Has the server been restarted?	Indicates whether the server has been rebooted during the last measurement period or not		The values reported by this measure and its corresponding numeric values are mentioned in the table below: Measure Value: No – Numeric value: 0 Measure Value: Yes – Numeric value: 1 **Note:** By default, this measure reports the above-mentioned **Measure Value**s to indicate whether the server has been rebooted or not. In the graph of this measure however, the Measure Values are represented using the numeric equivalents only. By checking the time periods when this metric changes from **No** to **Yes**, an administrator can determine the times when this server was rebooted. The Detailed Diagnosis of this measure, if enabled, lists the time, shutdown date, reboot date, shutdown duration, and reveals whether the server is in

Measure Value	Numeric value
No	0
Yes	1

Measurement	Description	Measurement Unit	Interpretation
			maintenance mode or not.
Total hits	Indicates the number of times the server was accessed by users during the last measurement period.	Number	
Data traffic	Indicates the rate at which data is transmitted by the web server during the last measurement period.	KB/Sec	
CPU usage	Indicates the prcentage of CPU utilized by the web server during the last measurement period.	Percent	If the value of this measure increases with time, it could be a cause for concern.
Requests rate	Indicates the rate at which HTTP requests were processed by the web server during the last measurement period.	Requests/sec	
Processing rate of requests	Indicates the amount of data transferred by the web server per second during the last measurement period.	KB/sec	
Average bytes per request	Indicates the amount of data transferred by the web server per HTTP request during the last measurement period.	KB/request	
Current requests	Indicates the number of HTTP requests that are currently being processed by the web server during the last measurement period.	Number	
Idle workers	Indicates the number of	Number	

Measurement	Description	Measurement Unit	Interpretation
	idle Apache processes waiting for an HTTP request during the last measurement period.		
Total server instances	Indicates the total number of server instances i.e., threads on the web server during the last measurement period.	Number	A very high value of this measure could indicate a processing bottleneck.
Instances waiting for connection	Indicates the total number of threads that were waiting for a connection during the last measurement period.	Number	
Currently starting instances	Indicates the number of HTTP requests that were currently processed by the web server during the last measurement period.	Number	
Instances reading requests	Indicates the number of HTTP requests that were reading the processes of the web server during the last measurement period.	Number	
Instances sending reply	Indicates the number of HTTP requests that received response from the web server during the last measurement period.	Number	
Instances in keepalive state	Indicates the number of processes in *KEEPALIVE* state in the web server during the last measurement period.	Number	
Instances in DNS lookup state	Indicates the number of requests for which DNS lookup state was enabled	Number	

Measurement	Description	Measurement Unit	Interpretation
	during the last measurement period.		
Instances in logging state	Indicates the number of processes that were in Logging state during the last measurement period.	Number	
Gracefully finishing instances	Indicates the number of processes that were in gracefully finishing state during the last measurement period.	Number	Sometimes after a graceful restart of the Apache server, certain extra processes i.e., the child processes are left running on the server and these processes are deemed to be in gracefully finishing state. These processes remain in this state indefinitely and will not go away until a stop-and-start or a restart is issued on the server. A low value is desired for this measure as frequent restarts to the server may gradually degrade the performance of the server.

3.2.3 HTTP Test

The details of the Http test that emulates a user accessing a web server are provided below. Since this test can be executed from a location external to the web server, this test presents an unbiased external perspective of the state of the web server.

Target of the test : An Apache web server

Agent deploying this test : An external agent; if you are running this test using the external agent on the eG manager box, then make sure that this external agent is able to communicate with the port on which the target Webserver is listening. Alternatively, you can deploy the external agent that will be running this test on a host that can access the port on which the target Web server is listening.

Outputs of the test : One set of outputs for every URL being monitored

Configurable parameters for the test

Parameters	Description
Test period	This indicates how often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port to which the specified host listens
URL	This test emulates a user accessing a specific web page(s) on the target web server to determine the availability and responsiveness of the server. To enable this emulation, you need to configure the test with the URLof the web page that it should access. Specify this URL against the URL parameter. If required, you can even configure multiple URLs – one each for every web page that the test should attempt to access. If each URL configured requires special permissions for logging in, then, you need to configure the test with separate credentials for logging into every URL. Likewise, you need to provide instructions to the test on how to validate the content returned by every URL, and also set an encoding format for each URL. To enable administrators to easily configure the above per URL, eG Enterprise provides a special interface. To access this interface, click on the encircled '+' button alongside the URL text box in the test configuration page. Alternatively, you can even click on the encircled '+' button adjacent to the URL parameter in the test configuration page. To know how to use this special interface, refer to Section **3.2.3.1**.
Cookiefile	Whether any cookies being returned by the web server need to be saved locally and returned with subsequent requests
ProxyHost	The host on which a web proxy server is running (in case a proxy server is to be used)
ProxyPort	The port number on which the web proxy server is listening
Proxyusername	The user name of the proxy server
Proxypassword	The password of the proxy server
Confirm password	Confirm the password by retyping it here.
Timeout	Here, specify the maximum duration (in seconds) for which the test will wait for a response from the server. The default Timeout period is 30 seconds.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Web Availability	This measurement indicates whether the	Percent	Availability failures could be caused by

Measurement	Description	Measurement Unit	Interpretation
	server was able to respond successfully to the query made by the test.		several factors such as the web server process(es) being down, the web server being misconfigured, a network failure, etc. Temporary unavailability may also occur if the web server is overloaded. Availability is determined based on the response code returned by the server. A response code between 200 to 300 indicates that the server is available.
Total response time	This measurement indicates the time taken by the server to respond to the requests it receives.	Secs	Response time being high denotes a problem. Poor response times may be due to the server being overloaded or misconfigured. If the URL accessed involves the generation of dynamic content by the server, backend problems (e.g., an overload at the application server or a database failure) can also result in an increase in response time.
TCP connection availability	This measure indicates whether the test managed to establish a TCP connection to the server.	Percent	Failure to establish a TCP connection may imply that either the web server process is not up, or that the process is not operating correctly. In some cases of extreme overload, the failure to establish a TCP connection may be a transient condition. As the load subsides, the server may start functioning properly again.
TCP connect time	This measure quantifies the time for establishing a TCP connection to the web server host.	Secs	Typically, the TCP connection establishment must be very small (of the order of a few milliseconds). Since TCP connection establishment is handled at the OS-level, rather than by the application, an increase in this value signifies a system-level bottleneck on the host that supports the web server.
Server response time	This measure indicates the time period between when the connection was	Secs	While the total response time may depend on several factors, the server response time is typically, a very good

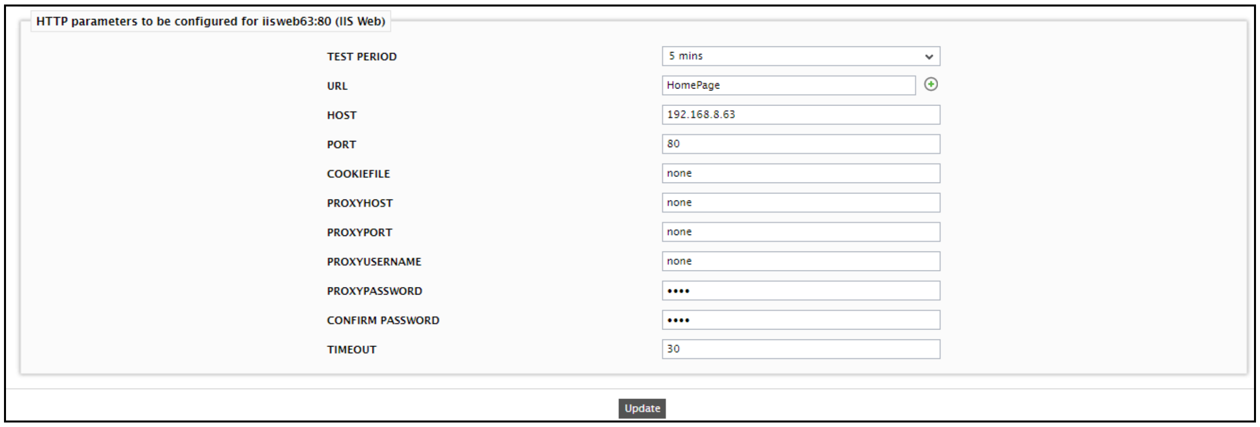
Measurement	Description	Measurement Unit	Interpretation
	established and when the server sent back a HTTP response header to the client.		indicator of a server bottleneck (e.g., because all the available server threads or processes are in use).
Response code	The response code returned by the server for the simulated request	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (eg., page not found). A 5xx value indicates a server error.
Content length	The size of the content returned by the server	Kbytes	Typically the content length returned by the server for a specific URL should be the same across time. Any change in this metric may indicate the need for further investigation on the server side.
Content validity	This measure validates whether the server was successful in executing the request made to it.	Percent	A value of 100% indicates that the content returned by the test is valid. A value of 0% indicates that the content may not be valid. This capability for content validation is especially important for multi-tier web applications. For example, a user may not be able to login to the web site but the server may reply back with a valid HTML page where in the error message, say, "Invalid Login" is reported. In this case, the availability will be 100 % (since we got a valid HTML response). If the test is configured such that the content parameter should exclude the string "Invalid Login," in the above scenario content validity would have a value 0.
DNS availability	Indicates whether the DNS server was able to respond successfully to the request made to it.	Percent	While the value 100 for this measure indicates that the DNS server is available and successfully responded to the request, the value 0 indicates that the DNS server is unavailable or is not responding to requests. Availability failures could be caused by many

Measurement	Description	Measurement Unit	Interpretation
			reasons such as a network failure. Sometimes, the DNS server may be reachable through basic network testing, but may not respond to DNS queries from clients. **Note:** This measure will be able to report a value only if the url parameter of the test is configured with a domain name-based URL – eg., http://www.eginnovations.com, http://www.eBooks.com. If the url parameter is configured with an IP-based URL instead – eg., http://192.168.10.21:80, http://192.168.10.34:7077 – then, this measure will not report any value. This is because, to report the availability of the DNS server, the test attempts to connect to the DNS server and resolve the domain name in the URL to its IP address. If the test is able to perform domain name – IP address resolution successfully, it reports the value 100 for this measure. If the resolution fails, the test reports the value 0. In case of an IP-based URL naturally, the test will not be able to find any domain name to resolve. The test therefore will not report any value for this measure in that case.
Data transfer time	Indicates the time taken for a data transfer between the drive and the host system.	Secs	Data transfer time being high denotes a problem.

3.2.3.1 Configuring Multiple URLs for Monitoring

By default, the HTTP test will be configured with the URL of the home page of the web server being monitored. To configure additional URLs, do the following:

1. Click on the encircled '+' button alongside the URL text box in Figure 3.4.

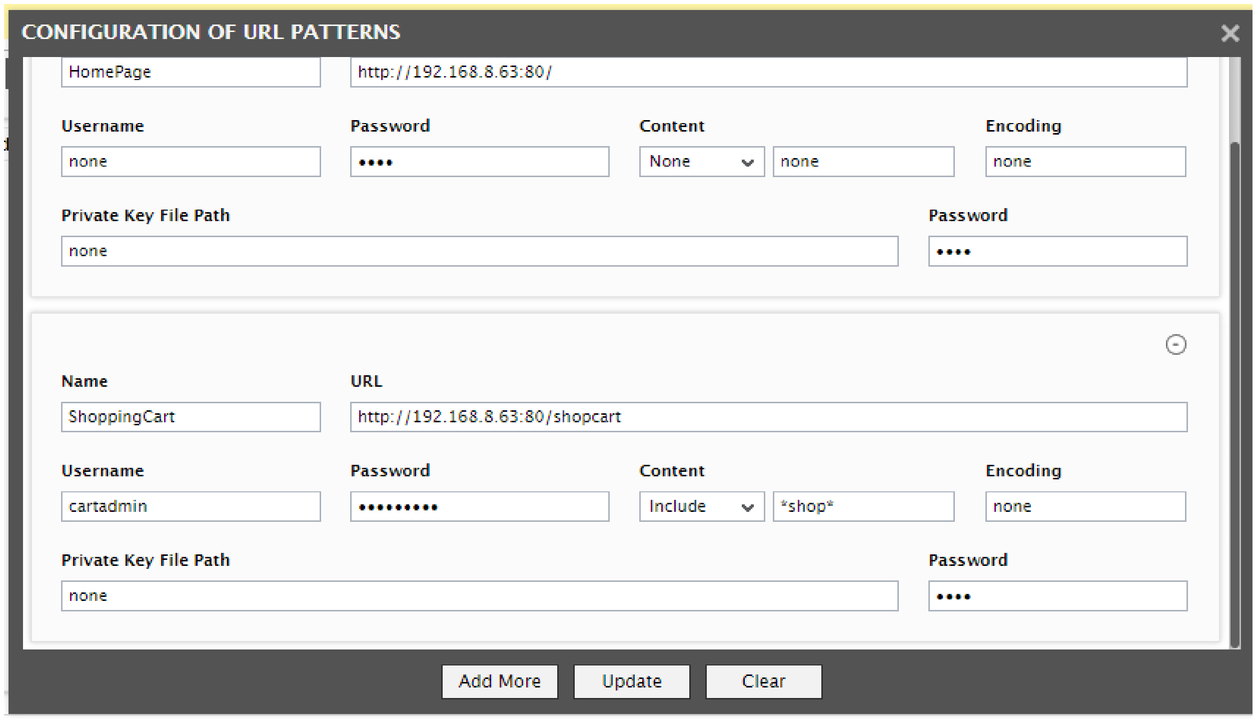


HTTP parameters to be configured for iisweb63:80 (IIS Web)

TEST PERIOD	5 mins
URL	HomePage
HOST	192.168.8.63
PORT	80
COOKIEFILE	none
PROXYHOST	none
PROXYPORT	none
PROXYUSERNAME	none
PROXYPASSWORD	••••
CONFIRM PASSWORD	••••
TIMEOUT	30

Figure 3.4: Configuring the HTTP test

2. Figure 3.5 then appears. To add another URL, click the **Add More** button in Figure 3.5.



CONFIGURATION OF URL PATTERNS

HomePage — http://192.168.8.63:80/

Username: none; Password: ••••; Content: None, none; Encoding: none

Private Key File Path: none; Password: ••••

Name: ShoppingCart; URL: http://192.168.8.63:80/shopcart

Username: cartadmin; Password: •••••••••; Content: Include, *shop*; Encoding: none

Private Key File Path: none; Password: ••••

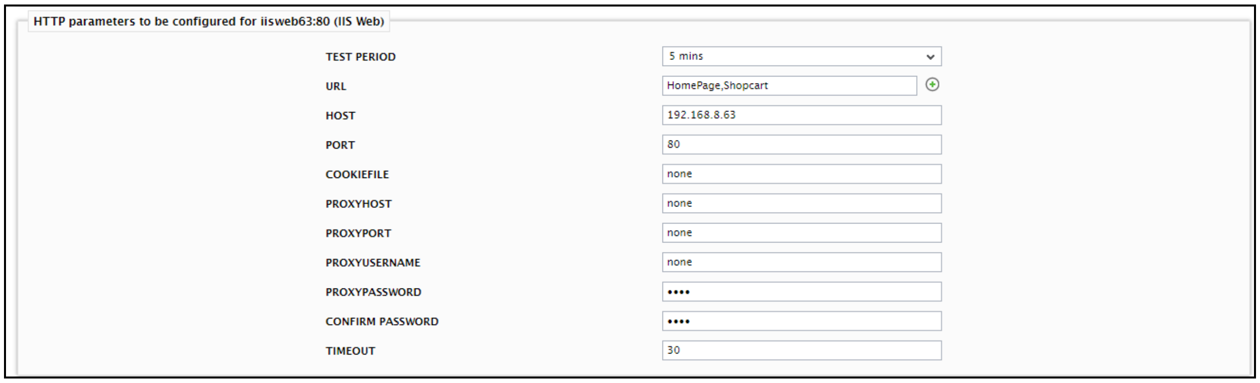
Figure 3.5: Configuring multiple URLs

3. Another URL specification section will appear. Specify the following in that section:

- **Name** : Specify a unique name by which the URL you will be specifiying shortly will be referred to across the eG user interface. This is the name that will appear as the descriptor of this test.
- **URL**: Enter the URL of the web page that this test should access.
- **Username** and **Password**: These parameters are to be set only if a specific user name / password has to be specified to login to the web page (i.e., **URL**) that you have configured for monitoring. In this case, provide valid login credentials using the **Username** and **Password** text boxes. If the web server on which **HTTP** test executes supports 'Anonymous user access', then these parameters will take either of the following values:
- A valid **Username** and **Password** for the configured **URL**
- *none* in both the **Username** and **Password** text boxes of the configured **URL**, if no user authorization is required
- Some web servers however, support NTLM (Integrated Windows) authentication, where valid login credentials are mandatory. In other words, a *none* specification will not be supported by such web servers. Therefore, in this case, against each configured **URL**, you will have to provide a valid **Username** in the format: domainname\username, followed by a valid **Password**.
- Please be sure to check if your web site requires HTTP authentication while configuring this parameter. HTTP authentication typically involves a separate pop-up window when you try to access the page. Many sites use HTTP POST for obtaining the user name and password and validating the user login. In such cases, the username and password have to be provided as part of the POST information and NOT as part of the *CREDENTIALS* specification for the **HTTP** test.
- **Content** : The **Content** parameter has to be configured with an instruction:value pair that will be used to validate the content being returned by the test. If the **Content** value is *None*, no validation is performed. On the other hand, if you pick the *Include* option from the **Content** list, it indicates to the test that for the content returned by the web server to be valid, the content must include the specified value (a simple string search is done in this case). This value should be specified in the adjacent text box. Similarly, if the **Exclude** option is chosen from the **Content** drop-down, it indicates to the test that the server's output is valid if it does not contain the value specified in the adjacent text box. The *Include* or *Exclude* value you specify in the text box can include wildcard characters. For example, an *Include* instruction can be *Home page*.

- **Encoding**: Sometimes the eG agent has to parse the *URL* content with specific encoding other than the default (ISO-8859-1) encoding. In such a case, specify the type of encoding using which the eG agent can parse the *URL* content in the **Encoding** text box. By default, this value is *none*.
- **Private Key File Path** and **Password**: SSL-enabled web sites are typically secured by a private key, public key, or a public-private key pair. If the web page configured for this test is SSL-enabled – i.e., if an HTTPS URL is specified against **URL** – and the contents of this web page can only be accessed using a private key, then the full path to the private key file will have to be provided against **Private key file path** and the password of the private key file should be specified against **Password**. If no such private key protects the contents of the configured **URL**, then set the **Private key file path** and its **Password** to *none*.

4. Similarly, you can add multiple URL specifications. To remove a URL specification, click on the encircled '-' button corresponding to it. To clear all URL specifications, click the **Clear** button in Figure 3.5. To update all the changes you made, click the **Update** button.
5. Once **Update** is clicked, you will return to the test configuration page (see Figure 3.6). The **URL** text box in the test configuration page will display just the **Name**s – i.e., the unique display names – that you may have configured for the multiple URLs, as a comma-separated list. To view the complete URL specification, click the encircled ''+' button alongside the **URL** text box, once again.


HTTP parameters to be configured for iisweb63:80 (IIS Web)

TEST PERIOD	5 mins
URL	HomePage,Shopcart
HOST	192.168.8.63
PORT	80
COOKIEFILE	none
PROXYHOST	none
PROXYPORT	none
PROXYUSERNAME	none
PROXYPASSWORD	••••
CONFIRM PASSWORD	••••
TIMEOUT	30

Figure 3.6: The URL text box displaying only the display names of the configured URLs as a comma-separated list

Chapter 4: Monitoring Web Sites/Web Tranactions on Apache Web Server Using eG web adapter capability

eG Enterprise offers a *Web* server model (see Figure 4.1) to monitor the Apache web servers, and pull out the statistics pertaining to the performance of the Apache web servers . The **Operating System**, **Network**, and **Tcp** layers have already been discussed in the *Monitoring Unix and Windows Servers* document. The **Application Processes** layer for the *Web* server is mapped to an additional **SSL Certificate Validity** test, which will be discussed later in this chapter. Above the **Application Processes** layer is the **Web Server** layer. This layer captures the state of the web server itself – whether the server is responding to user requests or not, whether the response is timely, whether the server is under overload, etc. Since a single web server may handle multiple web sites (e.g., in a virtual hosting scenario), the next layer is the **Web Site** layer that captures the health of a specific web site. While the overall health of a web site is represented by the **Web Site** layer, the status of the different key transactions of a web site are represented by the **Web Transactions** layer.

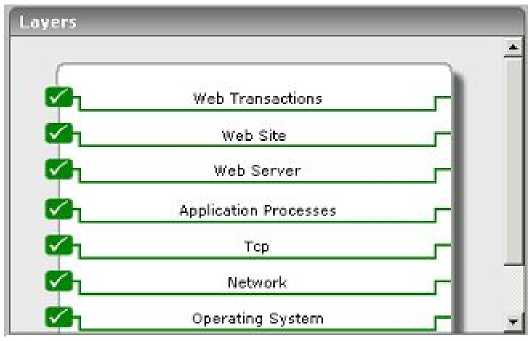


Figure 4.1: The different layers that the eG Enterprise suite monitors for an Apache web server

To monitor the health of Apache web servers, sites, and transactions, eG agents use the unique web adapter capability that allows a wealth of information to be collected from web servers without requiring explicit, expensive log files. More details of the web adapter technology can be found in the *eG Installation Guide* and the technical white papers.

The **Tcp**, **Network**, and **Operating System** layers have been discussed in great detail in the *Monitoring Unix and Windows Servers* document monitoring model . Likewise, **Application Processes** layer has already been discussed in the **Monitoring Apache Web Server** chapter.

4.1 The Web Server Layer

This layer tracks the health of a web server application. An external, Http test emulates a user connection to the web server, retrieves a web page from the server and measures the availability of the server and the time taken by the server to respond to the emulated user request. A second, internal WebServer test is based on eG Enterprise's web adapter capability (see Figure 4.2). The web adapter configured for a web server periodically monitors all the requests handled by the server. By snooping on user requests to the server and responses sent out by the server, the web adapter reports a variety of statistics pertaining to the web server. The WebServer test exports the information pertaining to the **Web Server** layer to the eG manager. The following sections describe the Http test and the **WebServer** test in detail.

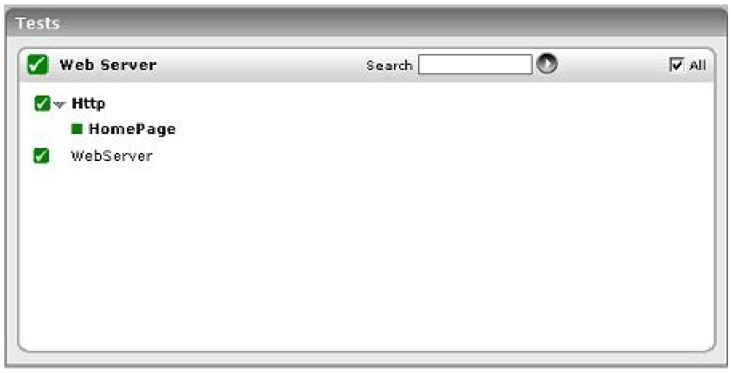


Figure 4.2: The tests that map to the Web Server layer

Since the **Http** test has been handled previously, the following sections will be dealing with the other tests.

4.1.1 Web Server Test

This internal test complements the measurements made from an external perspective by the **Http** test. In order to collect various statistics pertaining to the availability, performance, and usage of a web server, this test interfaces with the eG web adapter. The statistics collected by this test are detailed below:

Target of the test : An Apache web server

Agent deploying this test : An internal agent

Outputs of the test : One set of results for every web server monitored

Configurable parameters for the test

Parameters	Description
Test period	This indicates how often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port to which the specified host listens

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Connections	Rate of connections to the web server.	Conns/Sec	An increase or decrease in connection rate can represent a change in user workload
Requests	Rate of requests to the web server during the last measurement period.	Reqs/Sec	With the advent of HTTP/1.1, multiple requests can be transmitted over the same TCP connection. The ratio of requests per connection can provide an idea of the effectiveness of the HTTP 1.1 protocol.
Data transmitted	Rate at which the data was transmitted by the server during the last measurement period.	KB/Sec	A large increase in the data transmission rate can be indicative of an increase in the popularity of one or more web sites hosted on the server.
Data received	Rate at which the data was received by the server during the last measurement period.	KB/Sec	An increase in this value is indicative of an increase in user requests to the server.
Errors	Percentage of error responses from the server during the last measurement period.	Percent	Percentage of responses with a 400 or 500 status code.
Aborts	Percentage of requests aborted by users (a request is deemed to have been aborted if either the	Percent	A high percentage of aborts can happen if the web server’s responsiveness has dramatically reduced. Aborts may also occur because of backend errors (e.g.,

Measurement	Description	Measurement Unit	Interpretation
	connection is closed without any response being generated, or if the TCP connection is closed as the server is reading the request or as it is responding to the request) .		application failures, database connection problems etc.).
300 responses	Percentage of responses with a status code in the 300-399 range during the last measurement period.	Percent	300 responses could indicate page caching on the client browsers. Alternatively 300 responses could also indicate redirection of requests. A sudden change in this value could indicate a problem condition.
400 errors	Percentage of responses with a status code in the range 400-499 during the last measurement period.	Percent	A high value indicates a number of missing/error pages.
500 errors	Percentage of responses with a status code in the range 500-599 during the last measurement period.	Percent	Since responses with a status code of 500-600 indicate server side processing errors, a high value reflects an error condition.
Current requests	Number of server threads/processes currently in use for serving requests (this measurement is not available for Apache web servers).	Number	If a majority of the server threads/processes are in use simultaneously, this may be indicative of a server bottleneck.

Note:

The following measures are not available in the Windows version of the product:

- Aborts
- 300 responses
- 500 errors

This test will report metrics for an IIS web server executing on a Windows 2008 host, only if the **IIS Management Scripts and Tools** feature is installed on that host, and the **Web Server** role you create enables this feature. If the aforesaid feature is not enabled for the **Web Server** role, then remove the role and re-create it with the feature enabled.

4.1.2 HTTP Posts Test

The details of the HttpPost test are provided below. Since this test can be executed from a location external to the web server, this test presents an unbiased external perspective of the state of the web server. This test uses the POST command to submit its parameters. It is an optional test and can be configured only when the check box alongside it in the **DISABLED TESTS** list of the **AGENTS – TESTS CONFIGURATION** page is clicked. By default, this check box is deselected.

Target of the test : An Apache web server

Agent deploying this test : An external agent; if you are running this test using the external agent on the eG manager box, then make sure that this external agent is able to communicate with the port on which the target Webserver is listening. Alternatively, you can deploy the external agent that will be running this test on a host that can access the port on which the target Web server is listening.

Outputs of the test : One set of outputs for every URL being monitored

Configurable parameters for the test

Parameters	**Description**
Test period	This indicates how often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port to which the specified host listens
URL	The web page being accessed. While multiple URLs (separated by commas) can be provided, each URL should be of the format *URL name:URL value*. *URL name* is a unique name assigned to the URL, and the *URL value* is the value of the URL. For example, a URL can be specified as *HomePage:http://192.168.10.12:7077/*, where *HomePage* is the *URL name* and *http://192.168.10.12:7077/* is the *URL value*.
Cookiefile	Whether any cookies being returned by the web server need to be saved locally and returned with subsequent requests
ProxyHost	The host on which a web proxy server is running (in case a proxy server is to be used)
ProxyPort	The port number on which the web proxy server is listening

Parameters	Description
Proxyusername	The user name of the proxy server
Proxypassword	The password of the proxy server
Confirm password	Confirm the password by retyping it here.
Timeout	Here, specify the maximum duration (in seconds) for which the test will wait for a response from the server. The default Timeout period is 30 seconds.
Content	This is a set of *Instruction:Value* pairs that are used to validate the content being returned by the test. If the Content value is *none:none*, no validation is performed. The number of pairs specified in this text box, must be equal to the number of URLs being monitored. The instruction should be one of **Inc** or **Exc**. **Inc** tells the test that for the content returned by the web server to be valid, the content must include the specified value (a simple string search is done in this case). An instruction of **Exc** instructs the test that the server's output is valid if it does not contain the specified value. In both cases, the content specification can include wild card patterns. For example, an **Inc** instruction can be **Inc**: **Home page**.
Credentials	The **Http** test supports HTTP authentication. This parameter is to be set if a specific user name / password has to be specified to login to a page. This parameter is a comma separated list of user name:password pairs, one pair for each URL being monitored. A value of *none:none* indicates that user authorization is not required. Please be sure to check if your web site requires HTTP authentication while configuring this parameter. HTTP authentication typically involves a separate pop-up window when you try to access the page. Many sites use **HTTP POST** for obtaining the user name and password and validating the user login. In such cases, the username and password have to be provided as part of the **POST** information and **NOT** as part of the Credenials specification for the **Http** test.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Availability	This measurement indicates whether the server was able to respond successfully to the query made by the test.	Percent	Availability failures could be caused by several factors such as the web server process(es) being down, the web server being misconfigured, a network failure, etc. Temporary unavailability may also occur if the web server is overloaded. Availability is determined based on the response code returned

Measurement	Description	Measurement Unit	Interpretation
			by the server. A response code between 200 to 300 indicates that the server is available.
Total response time	This measurement indicates the time taken by the server to respond to the requests it receives.	Secs	Response time being high denotes a problem. Poor response times may be due to the server being overloaded or misconfigured. If the URL accessed involves the generation of dynamic content by the server, backend problems (e.g., an overload at the application server or a database failure) can also result in an increase in response time.
TCP connection availability	This measure indicates whether the test managed to establish a TCP connection to the server.	Percent	Failure to establish a TCP connection may imply that either the web server process is not up, or that the process is not operating correctly. In some cases of extreme overload, the failure to establish a TCP connection may be a transient condition. As the load subsides, the server may start functioning properly again.
TCP connect time	This measure quantifies the time for establishing a TCP connection to the web server host.	Secs	Typically, the TCP connection establishment must be very small (of the order of a few milliseconds). Since TCP connection establishment is handled at the OS-level, rather than by the application, an increase in this value signifies a system-level bottleneck on the host that supports the web server.
Server response time	This measure indicates the time period between when the connection was established and when the server sent back a HTTP response header to the client.	Secs	While the total response time may depend on several factors, the server response time is typically, a very good indicator of a server bottleneck (e.g., because all the available server threads or processes are in use).

Measurement	Description	Measurement Unit	Interpretation
Response code	The response code returned by the server for the simulated request	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (eg., page not found). A 5xx value indicates a server error.
Content length	The size of the content returned by the server	KBtes	Typically the content length returned by the server for a specific URL should be the same across time. Any change in this metric may indicate the need for further investigation on the server side.
Content validity	This measure validates whether the server was successful in executing the request made to it.	Percent	A value of 100% indicates that the content returned by the test is valid. A value of 0% indicates that the content may not be valid. This capability for content validation is especially important for multi-tier web applications. For example, a user may not be able to login to the web site but the server may reply back with a valid HTML page where in the error message, say, "Invalid Login" is reported. In this case, the availability will be 100 % (since we got a valid HTML response). If the test is configured such that the content parameter should exclude the string "Invalid Login," in the above scenario content validity would have a value 0.

4.1.3 SiteMinder Web Access Test

To control user accesses to a web site, web servers are configured to work with SiteMinder - a platform that authenticates, authorizes, and manages web sites. Using SiteMinder, administrators can enable Single Sign-On (SSO) for multiple web sites operating in an environment. With the SSO property of access control, a user logs in once and gains access to multiple web sites at one go, without being prompted to log into each of them separately. While SiteMinder’s SSO capability saves the time and effort required to manually sign into multiple web sites, a slowdown in SiteMinder can delay/suspend accesses to all the web sites managed (i.e., protected) by it! But should SiteMinder be blamed every time a user complaints of a slowdown when accessing a protected web site? Maybe not! Given below are the steps that occur when a user tries to access a web site protected by SiteMinder:

- The user requests for a web page using HTTP/HTTPS.
- The request is received by the web server and is intercepted by the SiteMinder web agent.
- The web agent determines whether or not the resource is protected. If the resource is protected, SiteMinder forces the user to login using their credentials. Typically, this is done via an HTTP POST request.
- SiteMinder authenticates the user and verifies whether or not the authenticated user is authorized for the requested web page, based on rules and policies specified in the Policy store.
- After the user is authenticated and authorized, SiteMinder grants access to the web page. Resource grant is done by providing a cookie to the client browser.

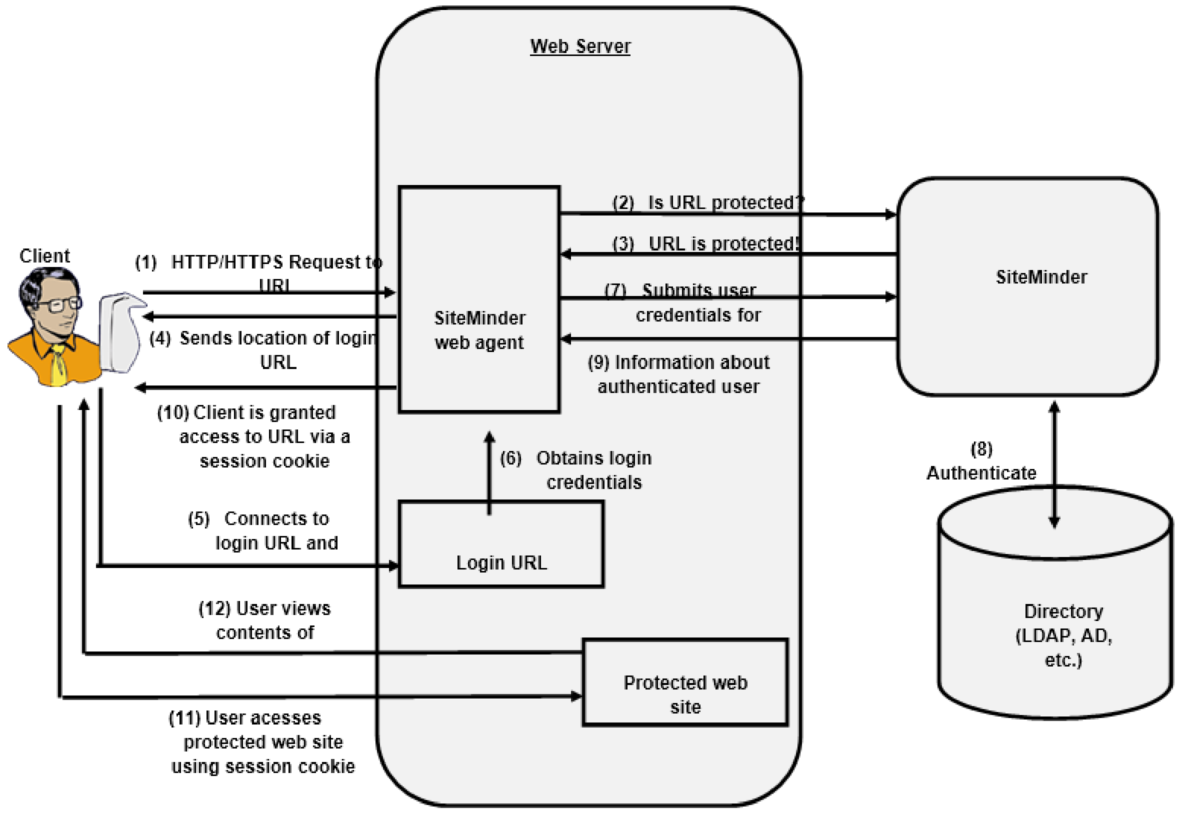

Web Server
Client
(1) HTTP/HTTPS Request to URL
(4) Sends location of login URL
(10) Client is granted access to URL via a session cookie
(5) Connects to login URL and
(12) User views contents of
(11) User accesses protected web site using session cookie
SiteMinder web agent
(2) Is URL protected?
(3) URL is protected!
(7) Submits user credentials for
(9) Information about authenticated user
(6) Obtains login credentials
Login URL
Protected web site
SiteMinder
(8) Authenticate
Directory (LDAP, AD, etc.)

Figure 4.3: Accessing a protected web site on a web server configured to work with SiteMinder

From Figure 4.3, it is clear that a problem in any step of this multi-step user interaction can delay web site accesses! This means that a slowdown in the SiteMinder policy server can be one of the reasons for a delay in accessing a web site, but it need not be the 'root-cause'! A processing bottleneck in the web server too can cause delays in web site accesses. Likewise, if the web agent takes too long to redirect the user's request to the login URL, the user experience with the site is bound to suffer. Similarly, if the web page requested is large and takes too long to load, access delays will become unavaoidable. In such circumstances, administrators will have trouble isolating the 'source' of the slowdown. If the root-cause of the slowdown is not diagnosed quickly, the problem will remain unresolved for a long time, resulting in a steep fall in service levels, increase in penalties and loss of reputation.

To avoid this, administrators will have to quickly pin-point the root-cause of a slowdown and fix it; for this, they need to know how much time each step of the user interaction with a protected web site takes. The *SiteMinder Web Access* test does this job. The test emulates a user accessing a web site protected by SiteMinder and reports the time taken by each step of the process, so that the precise step at which the slowdown occurred can be accurately isolated and the problem promptly eliminated.

Target of the test : An Apache web server

Agent deploying this test : An external agent; if you are running this test using the external agent on the eG manager box, then make sure that this external agent is able to communicate with the port on which the target Webserver is listening. Alternatively, you can deploy the external agent that will be running this test on a host that can access the port on which the target Web server is listening.

Outputs of the test : One set of outputs for every URL being monitored

Configurable parameters for the test

Parameters	Description
Test period	This indicates how often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port to which the specified host listens
URL	The web page being accessed. While multiple URLs (separated by commas) can be provided, each URL should be of the format *URL name:URL value*. URL name is a unique name assigned to the URL, and the URL value is the value of the URL. For example, a URL can be specified as *HomePage:http://192.168.10.12:7077/*, where HomePage is the URL name and http://192.168.10.12:7077/ is the URL value.

Parameters	Description
Cookiefile	Whether any cookies being returned by the web server need to be saved locally and returned with subsequent requests
ProxyHost	The host on which a web proxy server is running (in case a proxy server is to be used)
ProxyPort	The port number on which the web proxy server is listening
Proxyusername	The user name of the proxy server
Proxypassword	The password of the proxy server
Confirm password	Confirm the password by retyping it here.
Content	This is a set of *Instruction:Value* pairs that are used to validate the content being returned by the test. If the Content value is *none:none*, no validation is performed. The number of pairs specified in this text box, must be equal to the number of URLs being monitored. The instruction should be one of **Inc** or **Exc**. **Inc** tells the test that for the content returned by the web server to be valid, the content must include the specified value (a simple string search is done in this case). An instruction of **Exc** instructs the test that the server's output is valid if it does not contain the specified value. In both cases, the content specification can include wild card patterns. For example, an **Inc** instruction can be **Inc**: **Home page**. An **Inc** and an **Exc** instruction can be provided in quick succession in the following format: *Inc:*Home Page*,Exc:*home*.
Credentials	The parameter is to be set if a specific user name / password has to be specified to login to a page. Against this parameter, the *URLname* of every configured URL will be displayed; corresponding to each listed URLname, a Username text box and a Password text box will be made available. If the web server on which the test executes supports 'Anonymous user access', then this parameter will take either of the following values: • a valid Username and Password for every configured URLname • *none* in both the Username and Password text boxes of all configured URLnames (the default setting), if no user authorization is required
Timeout	Here, specify the maximum duration (in seconds) for which the test will wait for a response from the server. The default Timeout period is 30 seconds.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Availability	This measurement indicates whether or not this URL* was successfully accessed and valid content was returned. In other words, this measure clearly indicates whether or not step (1) to step (12) of Figure 4.3 completed successfully and the client was able to view the complete contents of the URL, till the last byte of data. * The term URL used in all discussions related to the measures of this test refers to the target URL indicated by step (1) of Figure 4.3, unless explicitly stated otherwise.	Percent	The value 100 for this measure indicates that the URL could be accessed successfully – i.e., the client could view the contents of the web page completely, till the last byte. The value 0 on the other hand indicates that the URL could not be accessed. This could be owing to a prolonged slowdown in the web server, performance issues with SiteMinder or an authentication failure reported by SiteMinder, processing bottlenecks experienced by the web agent, large size of the contents of the URL, etc. To zero-in on the exact reason for the inaccessibility of the configured URL, take a look at the values reported by the other measures of this test. **Note:** Even if the URL is not protected by SiteMinder, the 'Availability' measure will still report the value 100, provided the user does not receive any HTML response with a response code higher than 400. In this case, you can check the value reported by the 'URL SiteMinder protected' measure to figure out whether the URL is protected or not.
Total response time	This measurement indicates the total time it took for the client to request for a URL and receive access to that URL via a session cookie. This is the sum total of the time	Secs	Response time being high denotes a problem. Poor responsiveness can be caused due to a slowdown in the web server or SiteMinder, or because the content to be downloaded is large. Therefore, if the value of this measure

Measurement	Description	Measurement Unit	Interpretation
	taken to do the following: • The time taken by the client to know whether the URL is protected or not; • The time taken by the client to connect to the login URL and pass login credentials; • The time taken by the client to have his/her credentials authenticated by SitemInder; • The time taken by the client to view the complete contents of the requested URL, till its last byte; In short, this measure indicates the time taken to complete step (1) to step (12) of Figure 4.3 above.		is high or is increasing consistently, you will have to compare the values of the Web server response time, URL redirect time, Siteminder authentication time, and Content download time measures to figure out where the request spent maximum time; there is your bottleneck!
Web server availability	Indicates whether or not the web server is able to process the request for this URL and return the location of the login URL to the user. In other words, this measure indicates whether or not step (1) to step (4) of Figure 4.3 could be	Percent	The value 100 for this measure indicates that the web server is available, has successfully determined the protection status of the URL, and has returned the location of the login URL to the user. If the measure reports the value 0, it is indicative of the non-availability of the web server.

Measurement	Description	Measurement Unit	Interpretation
	completed successfully.		Availability is determined based on the response code returned by the server. If the server returns a response code between 200 and 300 the very first time the configured URL is hit, it indicates that the server is available. Any response code over 400 indicates non-availability of the web server. **Note:** Even if the URL is not protected by SiteMinder, the 'Web server availability' measure will still report the value 100, provided the user does not receive any HTML response with a response code higher than 400. In this case, you can check the value reported by the 'URL SiteMinder protected' measure to figure out whether the URL is protected or not.
Web server response time	Indicates the time taken to check and report whether this URL is protected or not; this is the time taken to perform steps (1), (2) and (3) of Figure 4.3.	Secs	Response time being high denotes a problem. Poor response times may be due to the web server being overloaded or misconfigured. If the value of the Total response time measure is high, then compare the value of the Web server response time measure with the other response time measures reported by this test to determine whether a processing bottleneck in the web server is the reason why the client had trouble accessing the web site.
URL redirect time	This measure indicates the time taken by the web agent on the web server to redirect the request to the login URL (indicated by	Secs	Ideally, the value of this measure should be low. A high value is indicative of poor responsiveness. If the value of the Total response time measure is high, then compare the

Measurement	Description	Measurement Unit	Interpretation
	step (5) of Figure 4.3); this is the total time taken to complete step (5) of Figure 4.3.		value of the URL redirect time measure with the other response time measures reported by this test to determine whether a delay in sending the location of the login URL to the client is what caused accesses to the web site to slow down.
URL SiteMinder protected	Indicates whether or not this URL is protected by SiteMinder; in other words, this measure checks whether the client received the login URL or not.	Boolean	While the value 1 for this measure indicates that the URL is protected by SiteMinder, the value 0 indicates that the URL is not protected.
SiteMinder authentication status	Indicates whether user authentication succeeded or failed.	Percent	A value of 100% indicates that SiteMinder successfully authenticated the login credentials that were submitted to it by the web agent. The value 0 on the other hand indicates an authentication failure. If the client received an sm_session cookie from the web agent, it indicates that authentication was successful. On the other hand, if the client did not receive the sm_session cookie it is indicative of authentication failure. The probable cause for this failure is the submission of invalid/incorrect credentials to SiteMinder.
SiteMinder authentication time	Indicates the time taken by SiteMinder to authenticate the user credentials; this is the time taken to perform steps (6), (7), (8), and (9) of Figure 2.4.	Secs	A low value is desired for this measure. A high value is indicative of an authentication bottleneck. If the value of the Total response time measure is high, then compare the value of the SiteMinder authentication time measure with the other response time measures reported by this test to determine whether a processing

Measurement	Description	Measurement Unit	Interpretation
			bottleneck in the SiteMinder Policy server is what is causing accesses to the web site to slow down.
Content validity	This measure validates whether the content returned by this URL is valid or not.	Percent	A value of 100% indicates that the content returned by the test is valid. A value of 0% indicates that the content may not be valid. Content that matches the value of the content parameter of this test is deemed as valid content; in this case, the measure will report the value 100. If the actual content does not match the content specification, then the value of the measure will be 0. This capability for content validation is especially important for multi-tier web applications. For example, a user may not be able to login to the web site but the server may reply back with a valid HTML page where in the error message, say, "Invalid Login" is reported. In this case, the availability will be 100 % (since we got a valid HTML response). If the test is configured such that the content parameter should exclude the string "Invalid Login," in the above scenario content validity would have a value 0.
Response code	Indicates the response code received by the client for the request to this URL.	Number	A value between 200 and 300 indicates a good response. A 4xx value indicates a problem with the requested content (eg., page not found). The value 401 is indicative of an authentication issue. A 5xx value indicates a server error.
Content length	The size of the content returned by the server for	Kbytes	Typically the content length returned by the server for a specific URL

Measurement	Description	Measurement Unit	Interpretation
	the request to this URL.		should be the same across time. Any change in this metric may indicate the need for further investigation on the server side.
Content download time	Indicates the total time taken by the client to view the complete contents returned by this URL till the last byte of data; in other words, this is the time taken to complete steps (11) and (12) of Figure 4.3.	Secs	A high value for this measure denotes that the content is taking a long time to load. This could be owing to the presence of a large number of images in the content or one/more large images. You may want to check the value of the Content length measure to understand how large the content is. If the value of the *Total response time* measure is high, then compare the value of the *Content download time* measure with the other response time measures reported by this test to determine whether the size of the content poses a road-block to swift access to the protected web site.

4.2 The Web Site Layer

The **Web Site** layer is specific to each web site hosted on a web server. An internal **WebSite** test shown in Figure 4.4 provides a comprehensive view of the states of the individual web sites supported by a web server.

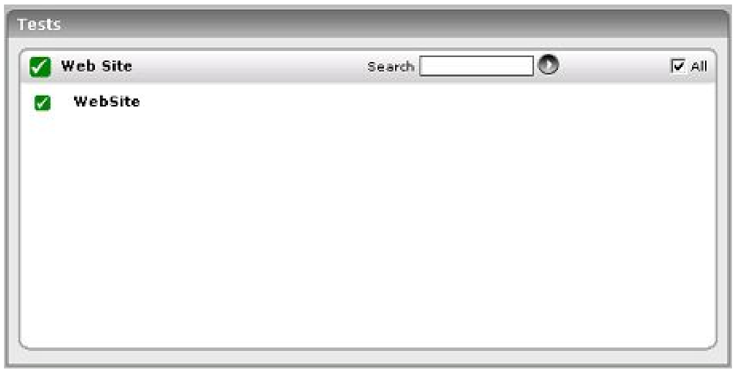


Figure 4.4: The WebSite test tracks the health of the Web Site layer

4.2.1 Web Site Test

Like the **WebServer** test, this test also interfaces with the eG web adapter to collect statistics relating to a web site.

Target of the test : An Apache web server

Agent deploying this test : An internal agent

Outputs of the test : One set of results for every web site that is being monitored

Configurable parameters for the test

Parameters	Description
Test period	This indicates how often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port to which the specified Host listens

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Connections	Rate of connections to the web site.	Conns/Sec	An increase or decrease in the connection rate can represent a change in the user workload.
Requests	Rate of requests to the web site.	Reqs/Sec	With the advent of HTTP/1.1, multiple requests can be transmitted over the same TCP connection. The ratio of requests per connection can provide an idea of the effectiveness of the HTTP 1.1 protocol.
Data transmitted	Rate at which the data is transmitted by the web site in response to user requests.	KB/Sec	A large increase in the data transmission rate can be indicative of an increase in the popularity of a web site hosted on the server.
Data received	Rate at which the data is received by the web site.	KB/Sec	An increase in this value is indicative of an increase in user requests to the web site.

Measurement	Description	Measurement Unit	Interpretation
Errors	Percentage of error responses from the web site.	Percent	Percentage of responses with a 400 or 500 status code.
Aborts	Percentage of requests aborted by users (a request is deemed to have been aborted if either the connection is closed without any response being generated, or if the TCP connection is closed as the server is reading the request or as it is responding to the request).	Percent	A high percentage of aborts can happen if the web site's responsiveness has dramatically reduced. Aborts may also occur because of backend errors (e.g., application failures, database connection problems etc.).
300 responses	Percentage of responses with a status code in the range 300-400.	Percent	Responses with a status code in the range 300-399 can indicate redirects from a server. Many a time, responses to cached content on the client also falls in this range.
400 errors	Percentage of responses with a status code in the range 400-500.	Percent	An unexpected increase in the percentage of responses with status codes in the range 400-499 can indicate a sudden problem at the server.
500 errors	Percentage of responses with a status code in the range 500-600.	Percent	Responses in the range 500-599 are caused by server-side errors (e.g., because of failures in the server side processing logic).
Current requests	Number of server threads/processes currently in use for serving requests for a web site (this measurement is not available for Apache web servers).	Number	A majority of the server threads/processes being used simultaneously to serve requests for a web site may be indicative of a server bottleneck caused by the web site.

Note:

The following measures are not available in the Windows version of the product:

- Aborts
- 300 responses
- 500 errors

4.3 The Web Transactions Layer

One of the unique capabilities of the eG Enterprise suite is its ability to monitor individual web transactions. The **WebSiteTransaction** test shown in Figure 4.5 is responsible for configuring an eG web adapter with the specifications of transactions corresponding to each web site being monitored. The web adapter is responsible for continuously tracking requests for these transactions and for periodically reporting a variety of statistics pertaining to each transaction back to the eG manager.

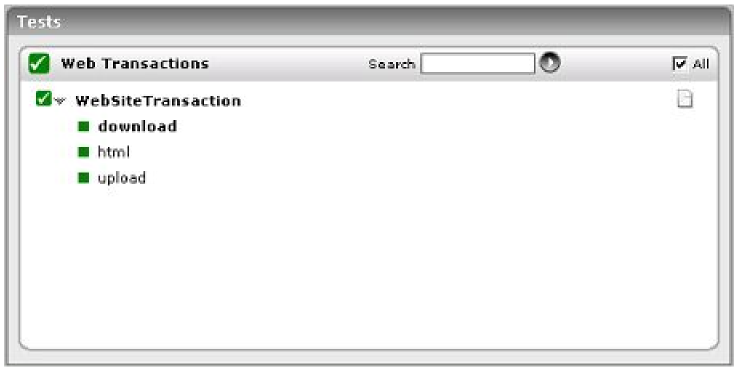


Figure 4.5: The tests that map to the Web Transactions layer of a web site.

4.3.1 Web Transactions Test

This test tracks the state of the individual transactions of a web site.

Target of the test : An Apache web server

Agent deploying this test : An internal agent

Outputs of the test : One set of results for each transaction supported by a web site

Configurable parameters for the test

Parameters	Description
Test period	This indicates how often should the test be executed.

Parameters	Description
Host	The host for which the test is to be configured.
Port	The port to which the specified host listens

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Requests	Rate of requests for a specific transaction.	Reqs/Sec	An increase or decrease in request rate for a specific transaction can represent a change in the user workload.
Errors	Percentage of error-filled responses from the web site for a specific transaction.	Percent	Percentage of responses for the transaction that report a 400 or 500 status code.
Aborts	Percentage of requests aborted by users when accessing a specific transaction from a web site (a request is deemed to have been aborted if either the connection is closed without any response being generated, or if the TCP connection is closed as the server is reading the request or as it is responding to the request).	Percent	A high percentage of aborts can happen if there has been a significant slow-down or error when users access a specific transaction (reasons for this could be errors in the application(s) supporting the transaction, failure of the backend database, etc.).
Data transmitted	Rate at which the data is transmitted by the web site in response to user requests for a specific transaction.	KB/Sec	A large increase in the data transmission rate can be indicative of an increase in the popularity of a specific transaction. Alternatively, a sudden increase in data rate may also indicate a change in the characteristics of a transaction.
Avg response time	The average time taken by the web site to respond to	Secs	An increase in response time is a major cause for user dissatisfaction with e-business sites. By correlating

Measurement	Description	Measurement Unit	Interpretation
	requests for a specific transaction, measured in seconds. Only requests for which successful responses are received are considered while computing the average response time.		an increase in response time with the other metrics collected by the agent per transaction, an operator can diagnose the reason(s) for the deterioration in performance.
Current requests	Number of server threads/processes currently in use for serving requests for the specific transaction supported by a web site (this measurement is not available for Apache web servers).	Number	If a majority of the server threads/processes are in use simultaneously to serve requests for a specific transaction, this may be indicative of a server bottleneck caused by the transaction being considered.

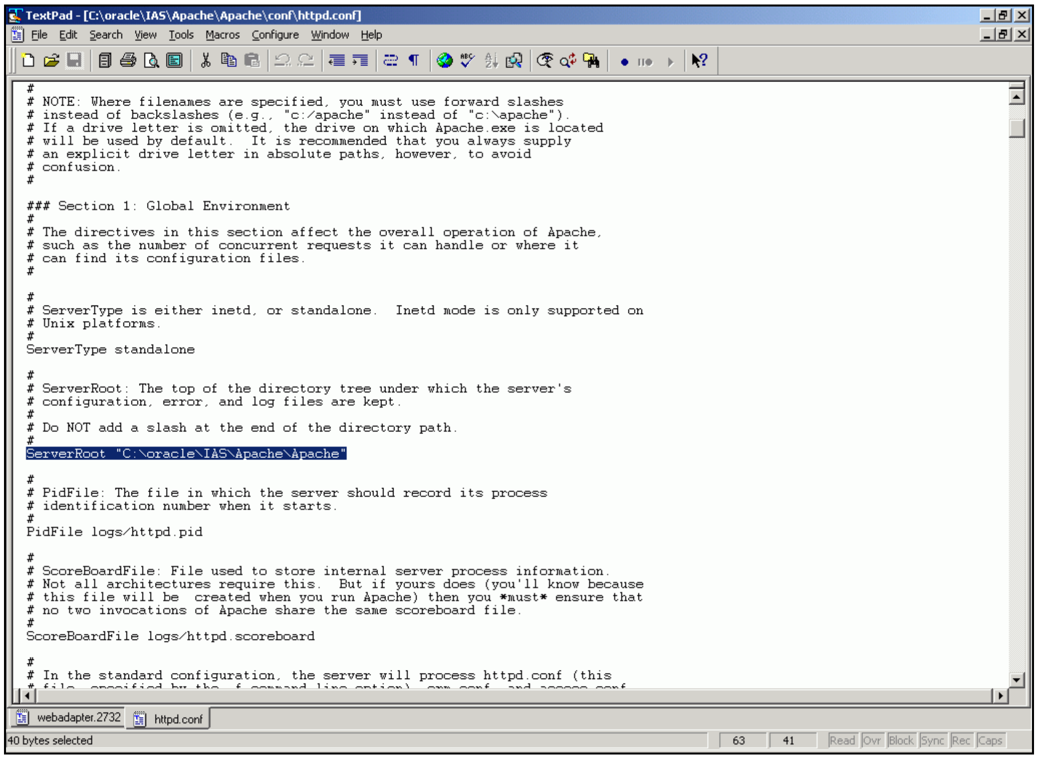
Note:

- For the Web Transactions test to run smoothly, logging will have to be enabled for the web site (s) being monitored. To enable logging, follow the procedure discussed in the *eG Installation Guide*.
- The *Aborts* and *Current requests* measures will not be available for Windows versions of the product.
- The Apache web server handles requests by spawning multiple threads/processes. As each thread/process handles requests, the eG instrumentation in the thread/process collects statistics for each request. Periodically, a summary of the new requests handled by a thread/process is updated to a common shared memory location from which the eG agent collects metrics and reports them to the eG manager. The frequency at which a thread/process updates the common shared memory is governed by an environment variable **REQUEST_INTERVAL**. By default, if this variable is not set, the eG web adapter uses a value of 5. That is, each thread/process updates the shared memory location once for every 5 requests that it handles. In a production system, with real user activity, this default setting will ensure that the overheads of the web adapter are minimal. In testing/staging environments, if the load on the web server is very low, the default **REQUEST_INTERVAL** setting may mean that the threads/processes do not update the shared memory location frequently. Hence, the administrator may notice slight discrepancies between the load imposed on the web server and the metrics reported by the web adapter. In such situations, the administrator can set the **REQUEST_INTERVAL** environment variable to 1 in

the Apache server’s start up script to force the Apache threads/processes to update the shared memory upon processing each request.

Chapter 5: Troubleshooting

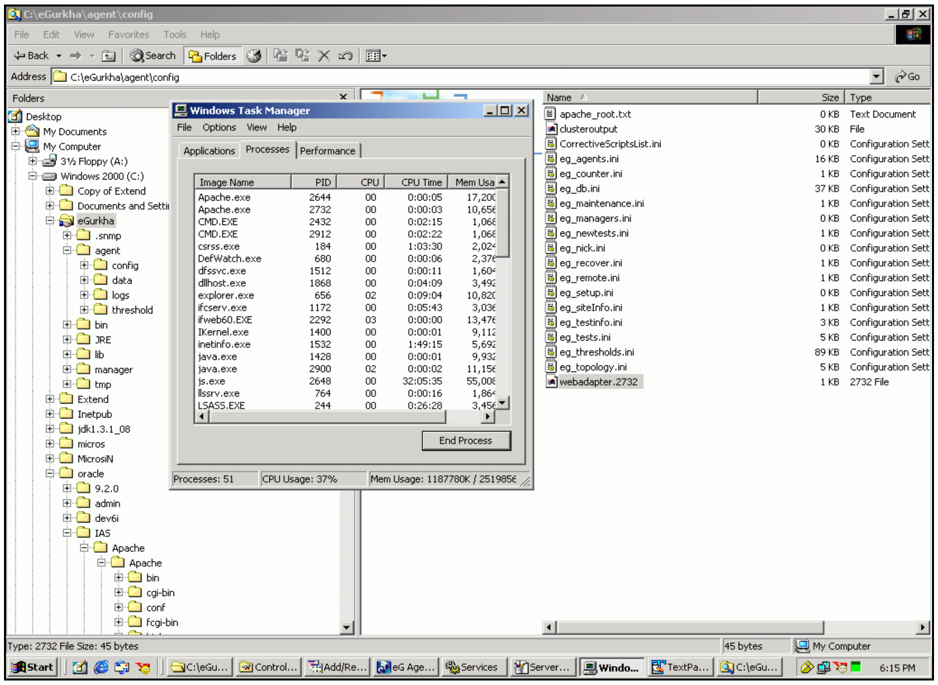
If the Apache server tests are in an **UNKNOWN** state, then proceed to check whether the web adapter has been configured properly. While configuring an Apache server, setup will request for the full path to the root directory of the server. Ensure that this path is the same as the value displayed against the **ServerRoot** parameter in the **HTTPd.conf** file in the <APACHE_SERVER_HOME >\conf directory (see Figure 5.1).


TextPad - [C:\oracle\IAS\Apache\Apache\conf\httpd.conf]
File Edit Search View Tools Macros Configure Window Help
webadapter.2732
httpd.conf
40 bytes selected
63
41
Read Ovr Block Sync Rec Caps

```
#
# NOTE: Where filenames are specified, you must use forward slashes
# instead of backslashes (e.g., "c:/apache" instead of "c:\apache").
# If a drive letter is omitted, the drive on which Apache.exe is located
# will be used by default.  It is recommended that you always supply
# an explicit drive letter in absolute paths, however, to avoid
# confusion.
#

### Section 1: Global Environment
#
# The directives in this section affect the overall operation of Apache,
# such as the number of concurrent requests it can handle or where it
# can find its configuration files.
#

#
# ServerType is either inetd, or standalone.  Inetd mode is only supported on
# Unix platforms.
#
ServerType standalone

#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# Do NOT add a slash at the end of the directory path.
#
ServerRoot "C:\oracle\IAS\Apache\Apache"

#
# PidFile: The file in which the server should record its process
# identification number when it starts.
#
PidFile logs/httpd.pid

#
# ScoreBoardFile: File used to store internal server process information.
# Not all architectures require this.  But if yours does (you'll know because
# this file will be  created when you run Apache) then you *must* ensure that
# no two invocations of Apache share the same scoreboard file.
#
ScoreBoardFile logs/httpd.scoreboard

#
# In the standard configuration, the server will process httpd.conf (this
```

Figure 5.1: The ServerRoot parameter in the HTTPd.conf file

Next, check whether a file named **webadapter.<PID>** is created in the <EG_ HOME_ DIR>\agent\config directory. This is a clear indicator of the successful deployment of the web adapter. Now, verify whether the **PID** in **webadapter.<PID>** matches with the **PID** of any one of the **Apache.exe** processes in the **Windows Task Manager** (see Figure 5.2). If it does not match, then the web adapter may not work. Under such circumstances, delete the **webadapter.<PID>** file and restart the Oracle web Server. Sometimes, an additional **webadapter** file will be created with a PID that does not match any of the **Apache.exe** PIDs listed in the **Windows Task Manager**. In such a case, delete the additional **webadapter.<PID>** file and restart the eG agent.



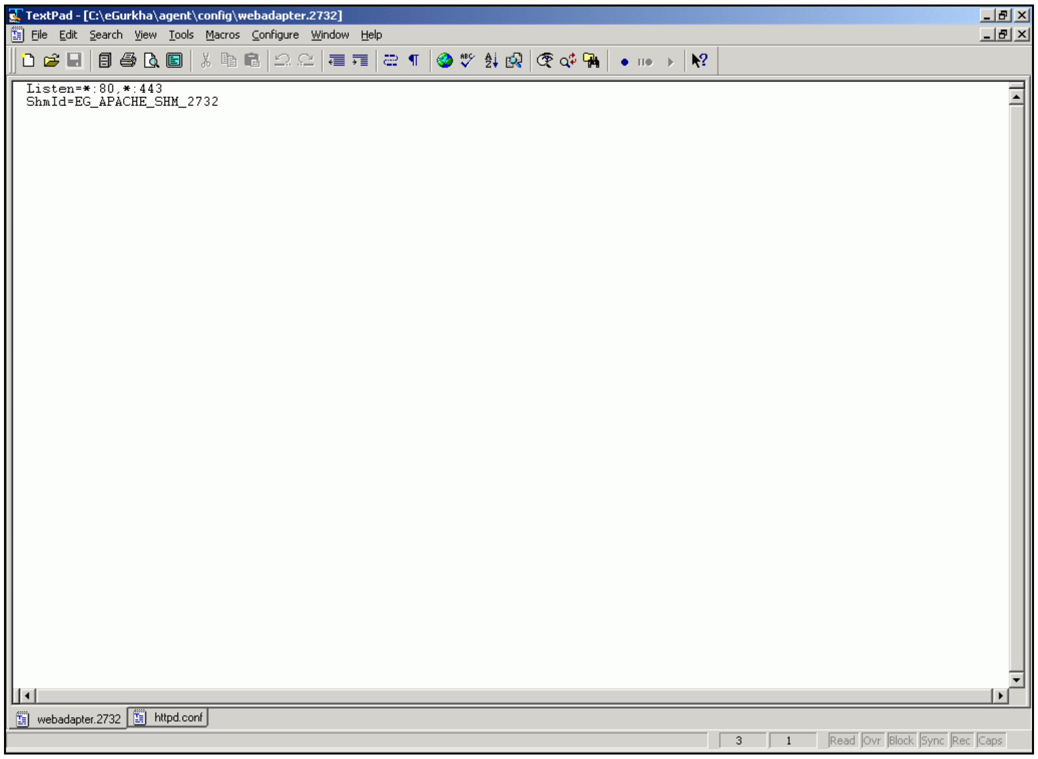
```
C:\eGurkha\agent\config
Address  C:\eGurkha\agent\config

Folders
Desktop
  My Documents
  My Computer
    3½ Floppy (A:)
    Windows 2000 (C:)
      Copy of Extend
      Documents and Setti
      eGurkha
        .snmp
        agent
          config
          data
          logs
          threshold
        bin
        JRE
        lib
        manager
        tmp
      Extend
      Inetpub
      jdk1.3.1_08
      micros
      MicrosiN
      oracle
        9.2.0
        admin
        dev6i
        IAS
          Apache
            Apache
              bin
              cgi-bin
              conf
              fcgi-bin
```

```
Windows Task Manager

Image Name        PID   CPU   CPU Time   Mem Usa
Apache.exe       2644   00    0:00:05    17,200
Apache.exe       2732   00    0:00:03    10,656
CMD.EXE          2432   00    0:02:15     1,068
CMD.EXE          2912   00    0:02:22     1,068
csrss.exe         184   00    1:03:30     2,024
DefWatch.exe      680   00    0:00:06     2,376
dfssvc.exe       1512   00    0:00:11     1,604
dllhost.exe      1868   00    0:04:09     3,492
explorer.exe      656   02    0:09:04    10,820
ifcserv.exe      1172   00    0:05:43     3,036
ifweb60.EXE      2292   03    0:00:00    13,476
IKernel.exe      1400   00    0:00:01     9,112
inetinfo.exe     1532   00    1:49:15     5,692
java.exe         1428   00    0:00:01     9,932
java.exe         2900   02    0:00:02    11,156
js.exe           2648   00   32:05:35    55,008
llssrv.exe        764   00    0:00:16     1,864
LSASS.EXE         244   00    0:26:28     3,456

Processes: 51   CPU Usage: 37%   Mem Usage: 1187780K / 2519856
```

```
Name                         Size   Type
apache_root.txt              0 KB   Text Document
clusteroutput               30 KB   File
CorrectiveScriptsList.ini    0 KB   Configuration Sett
eg_agents.ini               16 KB   Configuration Sett
eg_counter.ini               1 KB   Configuration Sett
eg_db.ini                   37 KB   Configuration Sett
eg_maintenance.ini           1 KB   Configuration Sett
eg_managers.ini              0 KB   Configuration Sett
eg_newtests.ini              1 KB   Configuration Sett
eg_nick.ini                  0 KB   Configuration Sett
eg_recover.ini               1 KB   Configuration Sett
eg_remote.ini                1 KB   Configuration Sett
eg_setup.ini                 0 KB   Configuration Sett
eg_siteInfo.ini              1 KB   Configuration Sett
eg_testinfo.ini              3 KB   Configuration Sett
eg_tests.ini                 5 KB   Configuration Sett
eg_thresholds.ini           89 KB   Configuration Sett
eg_topology.ini              5 KB   Configuration Sett
webadapter.2732              1 KB   2732 File

Type: 2732 File Size: 45 bytes   45 bytes   My Computer
```

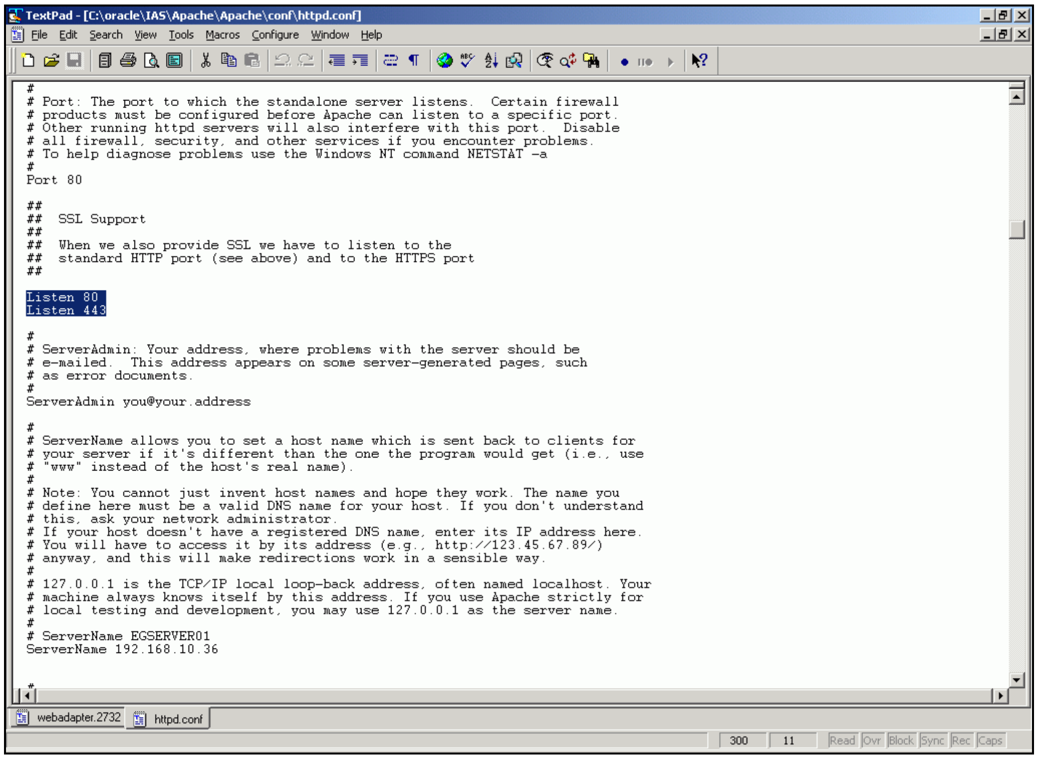
Figure 5.2: PID in the file name matching with the PID of one of the Apache.exe processes

Also, ensure that the **Listen** ports configured in the **webadpater.<PID>** file (see Figure 5.3) are the same as those which are listed in the **HTTPd.conf** file in the <APACHE_SERVER_HOME>\conf directory (see Figure 5.3).


TextPad - [C:\eGurkha\agent\config\webadapter.2732]
File Edit Search View Tools Macros Configure Window Help
webadapter.2732
httpd.conf

```
Listen=*:80,*:443
ShmId=EG_APACHE_SHM_2732
```

Figure 5.3: Listen ports displayed in the webadpater.<PID> file

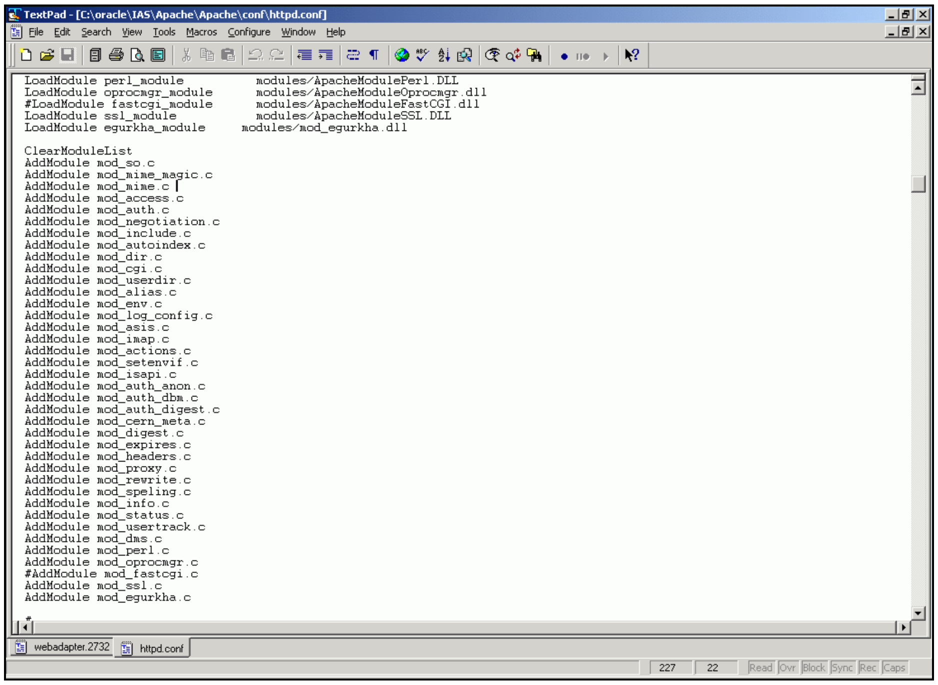

TextPad - [C:\oracle\IAS\Apache\Apache\conf\httpd.conf]
File Edit Search View Tools Macros Configure Window Help
webadapter.2732
httpd.conf

```
#
# Port: The port to which the standalone server listens.  Certain firewall
# products must be configured before Apache can listen to a specific port.
# Other running httpd servers will also interfere with this port.  Disable
# all firewall, security, and other services if you encounter problems.
# To help diagnose problems use the Windows NT command NETSTAT -a
#
Port 80

##
##  SSL Support
##
##  When we also provide SSL we have to listen to the
##  standard HTTP port (see above) and to the HTTPS port
##

Listen 80
Listen 443

#
# ServerAdmin: Your address, where problems with the server should be
# e-mailed.  This address appears on some server-generated pages, such
# as error documents.
#
ServerAdmin you@your.address

#
# ServerName allows you to set a host name which is sent back to clients for
# your server if it's different than the one the program would get (i.e., use
# "www" instead of the host's real name).
#
# Note: You cannot just invent host names and hope they work. The name you
# define here must be a valid DNS name for your host. If you don't understand
# this, ask your network administrator.
# If your host doesn't have a registered DNS name, enter its IP address here.
# You will have to access it by its address (e.g., http://123.45.67.89/)
# anyway, and this will make redirections work in a sensible way.
#
# 127.0.0.1 is the TCP/IP local loop-back address, often named localhost. Your
# machine always knows itself by this address. If you use Apache strictly for
# local testing and development, you may use 127.0.0.1 as the server name.
#
# ServerName EGSERVER01
ServerName 192.168.10.36
```

Figure 5.4: Listen ports displayed in the HTTPd.conf file

Note that the **Listen** ports displayed in the **webadapter.<PID>** file are prefixed by a '*', which typically represents an IP address. However, if a specific IP address substitutes the '*' in the **webadapter.<PID>** file, then, in the eG administrative interface, the Oracle web server must be managed using that IP address only.

Finally, check whether the directives indicated by Figure 5.5 exist in the **HTTPd.conf** file in the <APACHE_SERVER_HOME>\conf directory.


TextPad - [C:\oracle\IAS\Apache\Apache\conf\httpd.conf]
File Edit Search View Tools Macros Configure Window Help
webadapter.2732
httpd.conf
227
22
Read Ovr Block Sync Rec Caps

```
LoadModule perl_module            modules/ApacheModulePerl.DLL
LoadModule oprocmgr_module        modules/ApacheModuleOprocmgr.dll
#LoadModule fastcgi_module        modules/ApacheModuleFastCGI.dll
LoadModule ssl_module             modules/ApacheModuleSSL.DLL
LoadModule egurkha_module       modules/mod_egurkha.dll

ClearModuleList
AddModule mod_so.c
AddModule mod_mime_magic.c
AddModule mod_mime.c
AddModule mod_access.c
AddModule mod_auth.c
AddModule mod_negotiation.c
AddModule mod_include.c
AddModule mod_autoindex.c
AddModule mod_dir.c
AddModule mod_cgi.c
AddModule mod_userdir.c
AddModule mod_alias.c
AddModule mod_env.c
AddModule mod_log_config.c
AddModule mod_asis.c
AddModule mod_imap.c
AddModule mod_actions.c
AddModule mod_setenvif.c
AddModule mod_isapi.c
AddModule mod_auth_anon.c
AddModule mod_auth_dbm.c
AddModule mod_auth_digest.c
AddModule mod_cern_meta.c
AddModule mod_digest.c
AddModule mod_expires.c
AddModule mod_headers.c
AddModule mod_proxy.c
AddModule mod_rewrite.c
AddModule mod_speling.c
AddModule mod_info.c
AddModule mod_status.c
AddModule mod_usertrack.c
AddModule mod_dms.c
AddModule mod_perl.c
AddModule mod_oprocmgr.c
#AddModule mod_fastcgi.c
AddModule mod_ssl.c
AddModule mod_egurkha.c
```

Figure 5.5: eG-specific directives in the HTTPd.conf file

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.