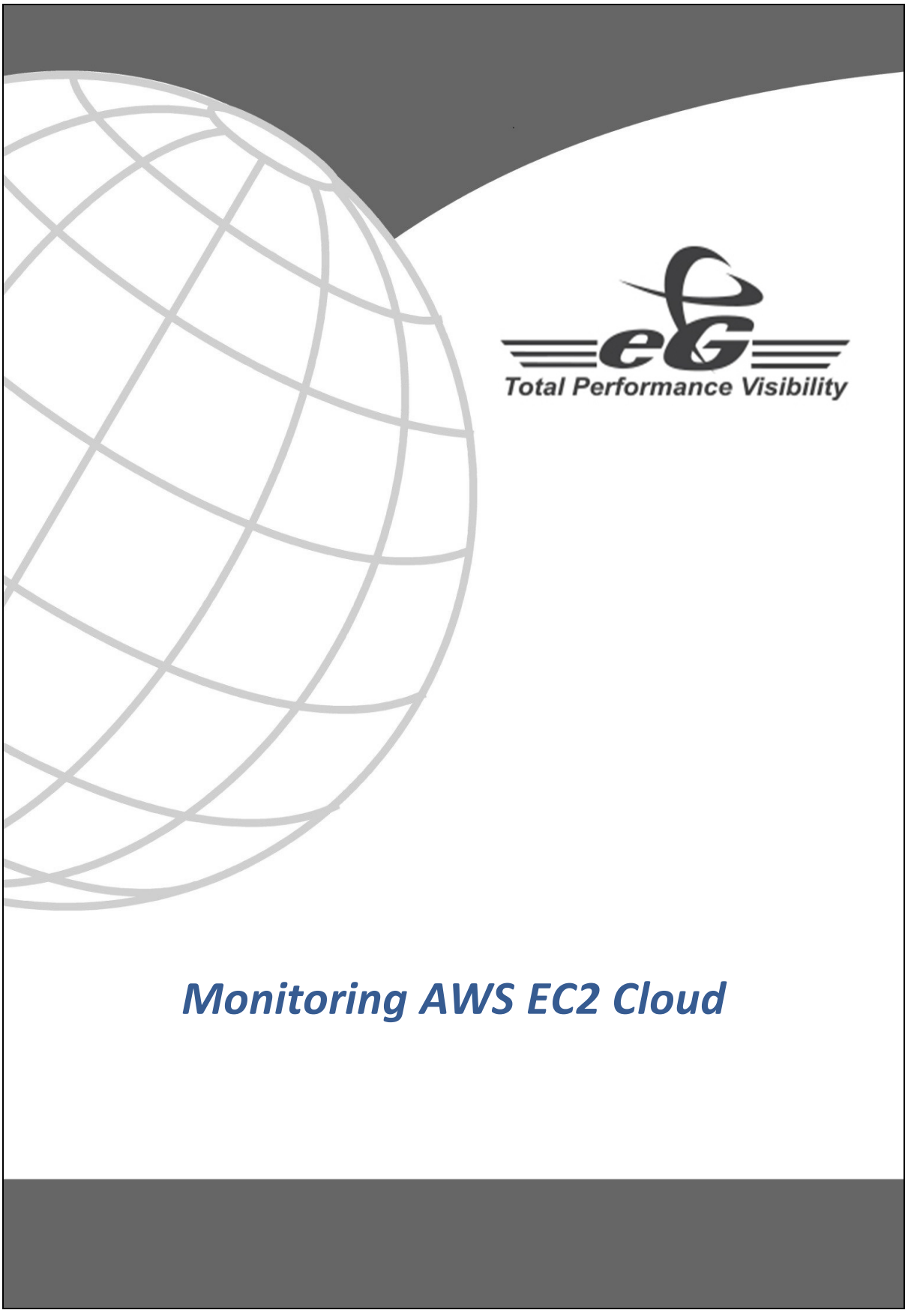


Monitoring AWS EC2 Cloud

Table of contents

Table of Figures

1

Introduction

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizeable computing capacity-literally, server instances in Amazon's data centers-that you use to build and host your software systems. You can get access to the infrastructure resources that EC2 provides by using APIs, or web tools and utilities.

With EC2, you use and pay for only the capacity that you need. This eliminates the need to make large and expensive hardware purchases, reduces the need to forecast traffic, and enables you to automatically scale your IT resources to deal with changes in requirements or spikes in popularity related to your application or service.

With many mission-critical applications now being delivered via the cloud, end-users have come to expect from the cloud the same quality of service that local service deployments are known to deliver. This means that even the slightest dip in performance levels will not be tolerated!

A sudden non-availability of the cloud, no matter how brief, or a slowdown/failure of any of its regions/availability zones/instances, can make it impossible for cloud providers to build and launch mission-critical services on the cloud and for consumers to access these services for prolonged periods. If you are a (public or private) cloud service provider therefore, your primary concerns would be - can people access my service? Is the self service portal up? Can users see their VMs? Can users connect to their VMs? If not, you need to be able to determine why the problem is happening - is it the web front-end? is it due to the virtualization platform? is it due to the SAN? etc. The action you take depends on what you diagnose as being the root-cause of the problem. Besides problem diagnosis, you are also interested in understanding how you can get more out of your current cloud investments. You want to be able to see how to balance load across your servers to serve a maximum number of users and how you can optimize the capacity of the infrastructure without sacrificing on performance. You need performance management “FOR” the cloud.

eG Enterprise is a unique solution that can provide you performance management FROM the cloud, OF the cloud and FOR the cloud!

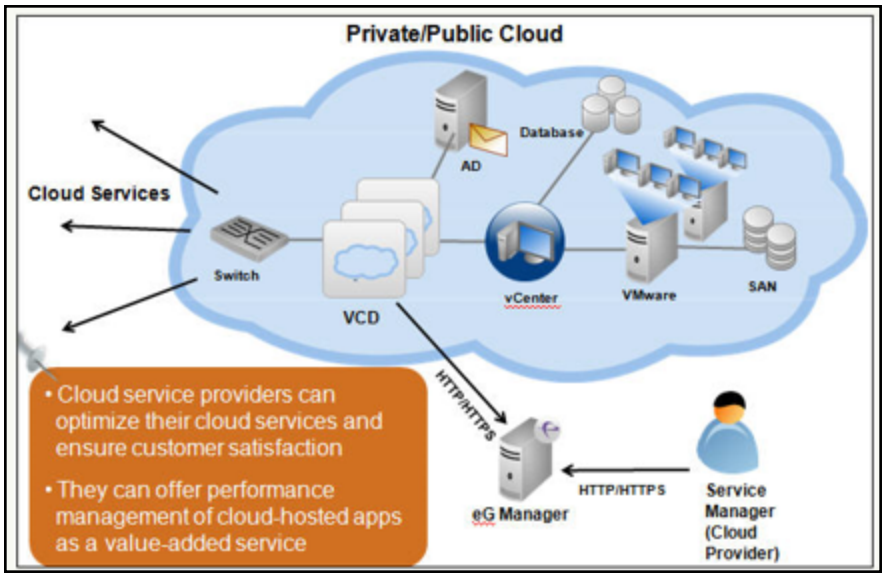


Figure 1.1: How eG monitors the cloud

To deliver performance management FOR the AWS EC2 cloud in particular, the solution offers two specialized monitoring models - the AWS EC2 Cloud model and the AWS EC2 Region model. The AWS EC2 Cloud monitoring model provides you with proactive updates on the overall health and status of the cloud and points you to unavailable regions/availability zones and resource-hungry instances in the cloud. To zoom into the health of specific regions and the instances operating within those regions, use the AWS EC2 Region model.

This document engages in detailed discussions on both the models.

2

Pre-Requisites for Monitoring the AWS EC2 Cloud Models

There are certain pre-requisites that are to be satisfied while you start monitoring the AWS EC2 Cloud models. These pre-requisites are discussed in detail in the forthcoming sections.

2.1 Configuring the eG agent with the required software

The main pre-requisite for an eG agent to be able to monitor the AWS EC2 models is to make sure that the eG agent monitoring the AWS EC2 models supports **JRE 1.6**. By default, JRE 1.5 is bundled with the eG agent on Linux and Windows(64-bit) platforms.

The AWS EC2 modelss are monitored in an *agentless* manner. Therefore, ensure that the remote agent supports JRE 1.6 by doing the following:

- Login to the agent host.
- Download JRE 1.6 for 64-bit hosts (if not already available on the host) from the **java.sun.com** web site.
- Then, copy the contents of the **jre** directory of JRE 1.6 (i.e., the **<JAVA_HOME>\jre directory**) to the **jre** directory of the eG agent (i.e., **<EG_AGENT_INSTALL_DIR>\ f**).
- Restart the eG agent.

2.2 Pricing details of the Amazon EC2 Instances

The Amazon instances are classified into six families namely Standard, Micro, High-Memory, High-CPU, Cluster Compute, and Cluster GPU.

Basically, configuring an instance within an Amazon EC2 Region and monitoring that instance is a **paid** service i.e., you pay charges for both configuring an instance in the Amazon EC2 cloud based on your requirement and collecting metrics from that particular instance. The payment varies according to the instance types and the Region in which the instance is deployed.

For more detailed information regarding the Amazon EC2 purchasing options refer http://aws.amazon.com/ec2/purchasing-options/

For more details regarding the pricing of the Amazon EC2 instances based on the Regions where they are deployed, refer http://aws.amazon.com/ec2/pricing/

2.3 Amazon Cloudwatch is a Paid service

Amazon CloudWatch is a web service that enables you to monitor your Amazon EC2 instances, in real-time. Metrics such as CPU utilization, latency, and request counts are provided automatically for these AWS

resources. You can also supply your own custom application and system metrics, such as memory usage, transaction volumes, or error rates, and Amazon CloudWatch will monitor these metrics too at a **nominal charge**. With Amazon CloudWatch, you can access up-to-the-minute statistics, view graphs, and set alarms for your metric data. Amazon CloudWatch functionality is accessible via API, command-line tools, the AWS SDK, and the AWS Management Console. Therefore, while configuring the tests for the AWS EC2 Component types, set the **CLOUDWATCH ENABLED** flag to **true**. By default, this flag is set to **true.**

2.4 Obtaining an Access key and Secret key

To monitor an Amazon EC2 instance, the eG agent has to be configured with the **access key** of a user with a valid AWS account. To obtain the **access key**, follow the steps given below:

- Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
- Provide the details of the user for whom you wish to create the AWS account.
- Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
- Once the payment is made, the user will automatically be signed in to the AWS account.
- From the newly created AWS account, you can request for an **access key**. You will be provided with an **access key** and a corresponding **secret key**.

While discovering an AWS EC2 Cloud component, provide the **access key** and **secret key** credentials that you have obtained in the corresponding text boxes.

Once the AWS EC2 Cloud is discovered, the Regions that are available within the Cloud will be discovered automatically.

3

Administering the eG Manager to Monitor the AWS EC2 Cloud

To achieve this, follow the steps given below:

1. Log into the eG administrative interface.
2. eG Enterprise automatically discovers the AWS EC2 Cloud component. If the AWS EC2 Cloud component is already discovered, use the Infrastructure -> Components ->Manage/Unmanage menu to manage it. Otherwise run discovery process as shown in Figure 3.1 from the menu sequence: Infrastructure ->Components -> Discovery. Provide the credentials that you had obtained while creating the AWS account. To know more about the AWS account, refer to Section 2.4 of this document.

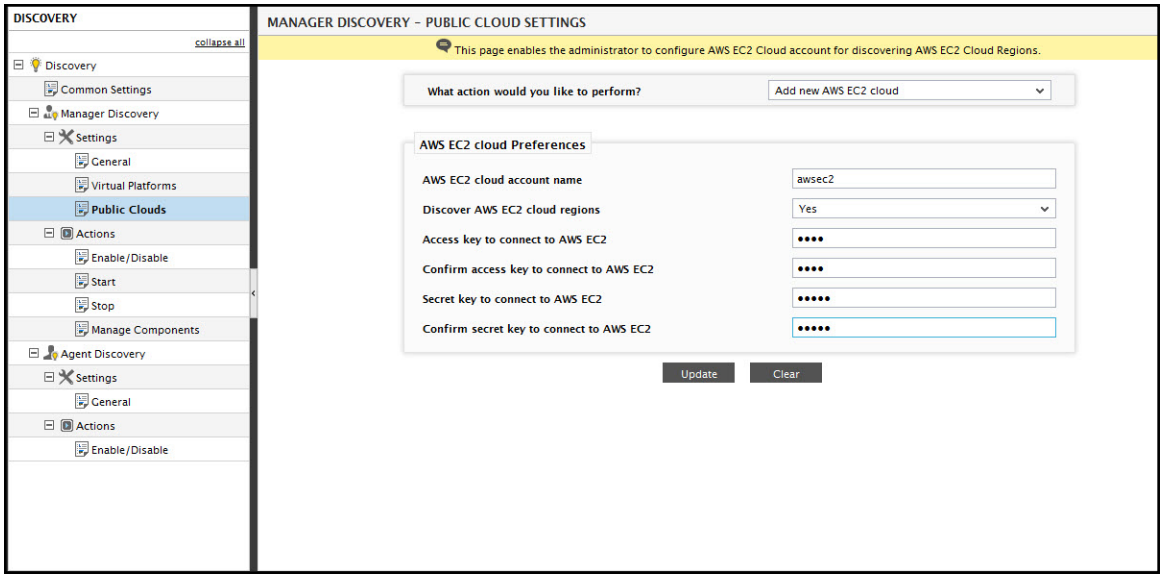


Figure 3.1: Providing the credentials during discovery of the AWS EC2 Cloud component

3. To manage the discovered components, go to the Infrastructure -> Components -> Manage/Unmanage page. The process of managing a component is clearly depicted by Figure 3.2 below.

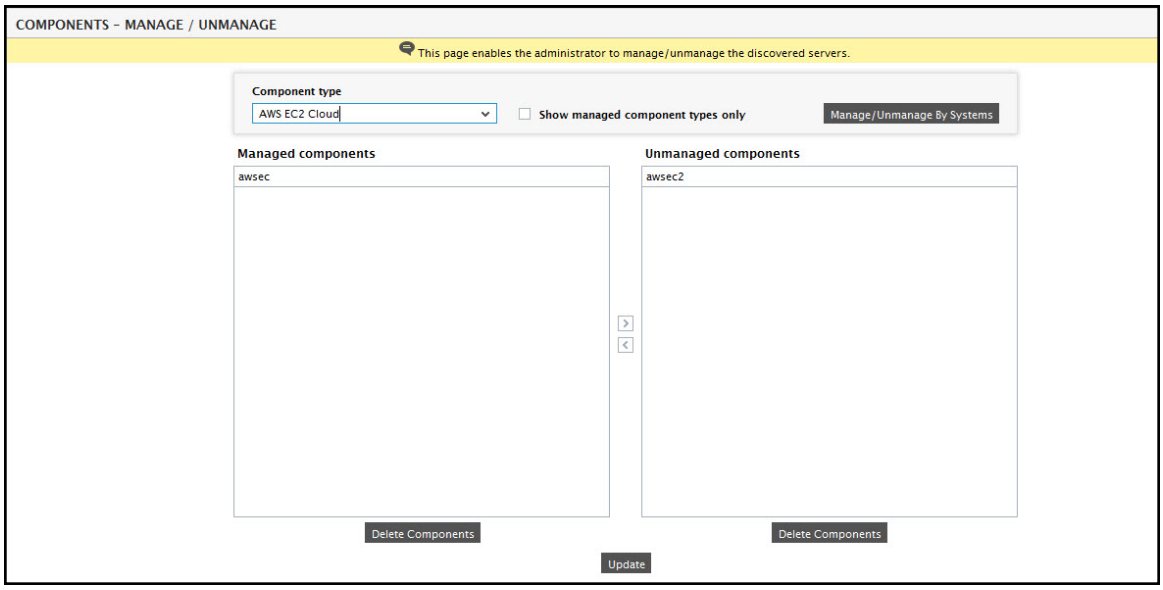


Figure 3.2: Managing the discovered AWS EC2 Cloud components

Note:

For a more detailed procedure for managing components, refer to **Configuring and Monitoring Web Servers** document.

4. You can also manually add the AWS EC2 Cloud component using Infrastructure -> Components -> Add/Modify. Remember that components manually added are managed automatically. While manually adding the AWS EC2 Cloud component, make sure that you provide a valid name in the HOST text box instead of an IP address. In order to provide a valid name, ensure that the **AllowQualifiedHostnames** flag is set to **Yes** in the **eg_services.ini** file of the **<EG_INSTALL_DIR>\manager\config** directory.
5. Now, when you attempt to sign out of the eG administrative interface, Figure 3.3 appears, listing the tests that require manual configuration.

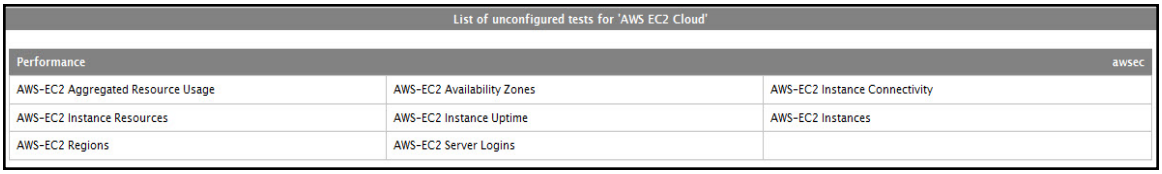


Figure 3.3: The list of unconfigured tests for AWS EC2 Cloud

6. Click on the **AWS – EC2 Regions** test to configure it. This test reports the availability of the default *Region* and enables the administrators to figure out the time taken by the default Region to respond to responses. To know how to configure the test, Click Here.
7. Once the test is configured, signout of the eG Administrative interface.

4

Monitoring the AWS EC2 Cloud

Figure 4.1 depicts the *AWS EC2 Cloud* monitoring model that eG Enterprise offers out-of-the-box for monitoring the Amazon EC2 cloud.

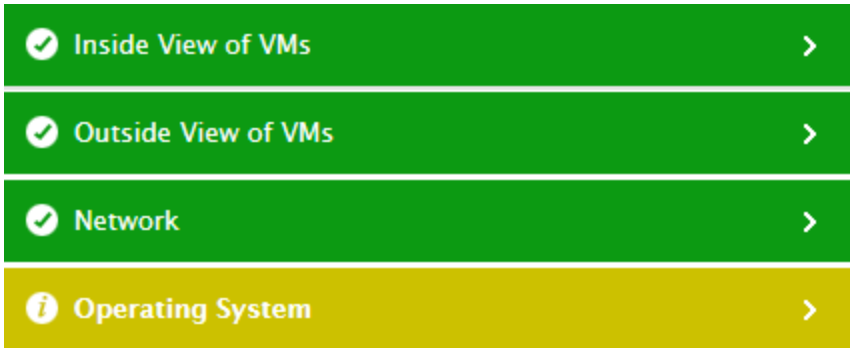


Figure 4.1: Layer model of the AWS EC2 Cloud

Each layer of this model is mapped to tests that reveal the availability of the cloud and whether the regions/availability zones/instances on the cloud are accessible. Using these statistics, cloud administrators can find quick and accurate answers for the following critical performance queries:

- Is web-based (HTTP/HTTPS) access to the cloud available?
- Does it take an unreasonably long time to establish contact with the cloud?
- How many regions does the cloud support? What are they?
- Is any region unavailable?
- Were any connectivity issues experienced while attempting to connect to a region? If so, which region is this?
- How many availability zones exist in each region? What are they?
- Is any availability zone currently unavailable? If so, which one is it?
- Is the default region on the cloud accessible? If so, is it taking too long to connect to the default region?
- Are all instances on the cloud accessible over the network?
- Are any instances powered off currently?
- Were any instances launched/removed recently? If so, which ones are these?
- What type of instances are resource-intensive?
- Is any particular instance consuming too much CPU?
- Is the network traffic to/zofrom any instance unusually high?
- Is the disk I/O of instances optimal?
- Was any instance rebooted recently? If so, which one is it?

Note:

The eG agent reports metrics for only those regions, availability zones, and instances on the cloud that the configured AWS user account is allowed to access.

Some tests require the **AWS CloudWatch** service to be enabled. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. For enabling this service, you need to pay CloudWatch fees. Refer to the AWS web site for the fee details.

The sections that will follow discuss each of the layers of Figure 4.1 in great detail.

4.1 The AWS Infrastructure Layer

Using the tests mapped to this layer, you can promptly detect the non-availability of the cloud, inaccessibility of regions and availability zones on the cloud, and connection bottlenecks experienced while connecting to the cloud or its components.

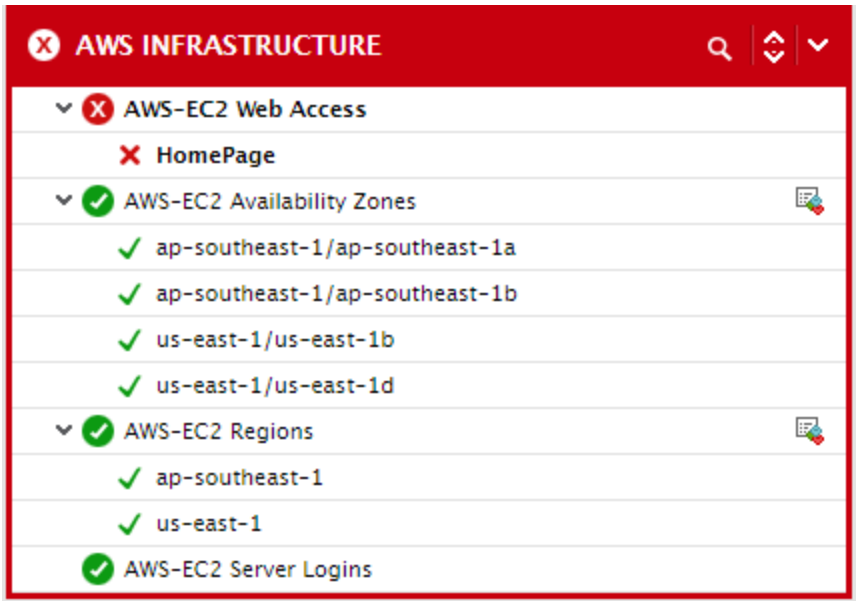


Figure 4.2: The test associated with the AWS Infrastructure layer

4.1.1 AWS-EC2 Web Access Test

This test emulates a user accessing a web page on the cloud via HTTP(S), and reports whether that page is accessible or not. In the process, the test indicates the availability of the cloud over the web, and the time it took for the agent to access the cloud over the web. This way, issues in web-based access to the cloud come to light.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for every URL configured for monitoring

Test parameters:

1. **TEST PERIOD** – How often should the test be executed
2. **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. By default, the url parameter is set to *HomePage:http://aws.amazon.com/ec2/* , where *HomePage* is the **URL name** , and *http://aws.amazon.com/ec2*is the **URL** value. You can modify this default setting to configure any **URL** of your choice - eg., the URL of the login page to your cloud-based infrastructure.
3. **HOST** - The host for which the test is to be configured.
4. **PORT** - The port to which the specified **HOST** listens
5. **COOKIEFILE** – Whether any cookies being returned by the web server need to be saved locally and returned with subsequent requests
6. **PROXYHOST** – The host on which a web proxy server is running (in case a proxy server is to be used)
7. **PROXYPORT** – The port number on which the web proxy server is listening
8. **PROXYUSERNAME** – The user name of the proxy server
9. **PROXYPASSWORD** – The password of the proxy server
10. **CONFIRM PASSWORD**– Confirm the password by retyping it here.
11. **CONTENT** – 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 **URL**s being monitored. The instruction should be one of **Inc** or **Exc**. **Inc** tells the test that for the content returned by the test 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 test'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*.
12. **CREDENTIALS**– The HttpTest supports HTTP authentication. The **CREDENTIALS** 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. These parameters will take either of the following values:
    - 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

    Where NTLM (Integrated Windows) authentication is supported, valid CREDENTIALS are mandatory. In other words, a *none* specification will not be supported in such cases. Therefore, in this case,

against each configured URLname, 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 this test.

13. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
13. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none** , indicating that the proxy sever does not require authentication by default.
14. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
15. **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.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Availability:	Indicates whether the test was able to access the configured URL or not	Percent	Availability failures could be caused by several factors such as the web server process(es) (hosting the configured web page) 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 test. A response code between 200 to 300 indicates that the configured web page is available.
Total response	Indicates the time taken	Secs	Response time being high denotes a problem.

Measurement	Description	Measurement Unit	Interpretation
time:	by the test to access this URL		Poor response times may be due to an overload. If the URL accessed involves the generation of dynamic content, 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:	Indicates whether the test managed to establish a TCP connection to this URL.	Percent	Failure to establish a TCP connection may imply that either the web server process hosting the web page 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 web page may start functioning properly again.
Tcp connect time:	Quantifies the time for establishing a TCP connection to the configured URL.	Secs	Typically, the TCP connection establishment must be very small (of the order of a few milliseconds).
Server response time:	Indicates the time period between when the connection was established and when the test 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).
Response code:	Returned by the test 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 test.	Kbytes	Typically the content length returned by the test for a specific URL should be the same across time. Any change in this metric may indicate the need for further investigation.

Measurement	Description	Measurement Unit	Interpretation
Content validity:	Validates whether the test 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.

4.1.2 AWS-EC2 Availability Zones Test

Amazon has data centers in different areas of the world (e.g., North America, Europe, Asia, etc.). Correspondingly, EC2 is available to use in different *Regions*. Each Region contains multiple distinct locations called *Availability Zones* (illustrated in the following diagram). Each Availability Zone is engineered to be isolated from failures in other Availability zones and to provide inexpensive, low-latency network connectivity to other zones in the same Region. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location.

If users complaint that their server instances are inaccessible, you may want to know whether it is because of the non-availability of the availability zone within which the instances have been launched. This test auto-discovers the regions and availability zones on the Amazon EC2 Cloud, and reports the availability of each zone.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each availability zone in each region of the AWS EC2 Cloud being monitored

First-level descriptor: AWS Region

Second-level descriptor: Availability zone

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and aws-ec2 vm Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees**. **For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use,

then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

10. **REPORT INSTANCE DATACENTER** - By default, this test reports the availability of only those availability zones that contain one/more instances. Accordingly, this flag is set to **true** by default. If you want the test to report metrics for all availability zones, regardless of whether/not they host instances, set this flag to **false**.
11. **EXCLUDE INSTANCE** - **This parameter applies only to AWS-EC2 Instance Connectivity, AWS-EC2 Instance Resources , and AWS-EC2 Instance Uptime tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.
12. **DETAILED DIAGNOSIS**- To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Availability:	Indicates whether/not this availability zone in this region is currently available.	Number	The value *0* indicates that the availability zone is *Not Available* and the value *100* indicates that it is *Available*. If an availability zone fails, then all server instances operating within that zone will also be rendered unavailable. If you host all your Amazon EC2 instances in a single location that is affected by such a failure, your instances will be unavailable, thereby bringing your entire application to a halt. On the other hand, if you have instances distributed across many Availability Zones and one of the instances fails, you can design your application so the instances in the remaining Availability Zones handle any requests.

4.1.3 AWS-EC2 Regions Test

Amazon EC2 provides the ability to place instances in multiple locations. Amazon EC2 locations are composed of Availability Zones and Regions. Regions are dispersed and located in separate geographic areas (US, EU, etc.). Each Region is completely independent.

By launching instances in separate Regions, you can design your application to be closer to specific customers or to meet legal or other requirements.

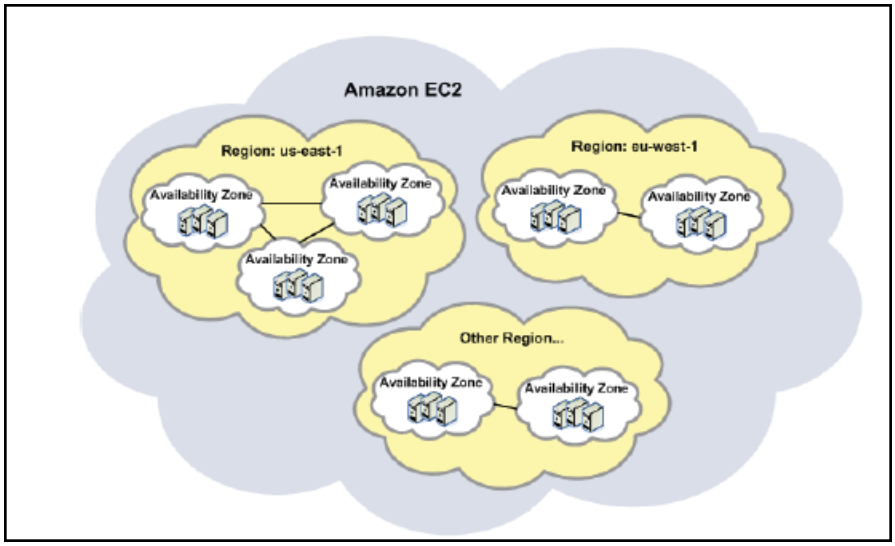


Figure 4.3: Regions and Availability zones

If a region is unavailable, then users to that region will not be able to access the server instances launched in that region. This may, in turn, adversely impact the user experience with the cloud. To avoid such an unpleasant outcome, it is best to periodically monitor the availability of each region, so that unavailable regions can be quickly and accurately identified, and the reasons for their non-availability remedied.

This test performs periodic availability checks on each region on the cloud, and reports the status of the individual regions. In addition, the test also indicates the time taken for connecting to a region so that, regions with connectivity issues can be isolated.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each region of the AWS EC2 Cloud being monitored

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured

3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and aws-ec2 vm Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees. For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

10. **REPORT INSTANCE REGION** - By default, this test reports metrics for only those regions that host at least one instance. This is why, the **REPORT INSTANCE REGION** flag is set to **Yes** by default. If you want, you can configure this test to report metrics for all regions, regardless of whether/not they host any instances. For this, set this flag to **No**.
11. **EXCLUDE INSTANCE** - **This parameter applies only to AWS-EC2 Instance Connectivity, AWS-EC2 Instance Resources , and AWS-EC2 Instance Uptime tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Availability:	Indicates whether/not this region is currently available.	Number	The value 0 indicates that the region is Not Available and the value 100 indicates that it is Available.
Response time:	Indicates the time taken to connect to this region.	Secs	A low value is typically desired for this measure. A high value or a consistent increase in this value could be indicative of connection bottlenecks. Compare the value of this measure across regions to know which region takes the longest to connect to.

4.1.4 AWS-EC2 Server Logins Test

This test attempts to connect to the default region in the cloud; in the process, the test reports whether the configured AWS user account is able to access the cloud-based infrastructure or not, and if so, how quickly the connection with the infrastructure was established.

If a user is denied access to a server instance on a clod, or if a user experiences a significant delay in connecting to his/her instances, you can use this test to validate the user credentials and to figure out whether any connectivity issues exist.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for the AWS EC2 Cloud being monitored

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and aws-ec2 vm Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees. For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation

name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

10. **EXCLUDE INSTANCE** - **This parameter applies only to AWS-EC2 Instance Connectivity, AWS-EC2 Instance Resources , and AWS-EC2 Instance Uptime tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Default region availability:	Indicates whether or not the test is able to access the default region on the cloud using the configured AWS user account .	Percent	The value 0 indicates that the region is not accessible, and the value 100 indicates that it is accessible. If the default region is inaccessible, it could be owing to any one of the following reasons: • The cloud is unavailable; • The configured AWS account does not have the access rights to the default region; • The test has been configured with incorrect login credentials.
Response time:	Indicates the time taken by the test to establish a connection with the default region on the cloud.	Secs	A low value is desired for this measure. A high value or a consistent increase in this value could indicate connection bottlenecks.

4.2 The AWS Cloud Instances Layer

The tests mapped to this layer take stock of the total number of instances (that are available for the configured AWS user account) on the cloud, and points you to the following:

- The powered-off instances
- The newly launched/removed instances
- Instances that are unavailable over the network

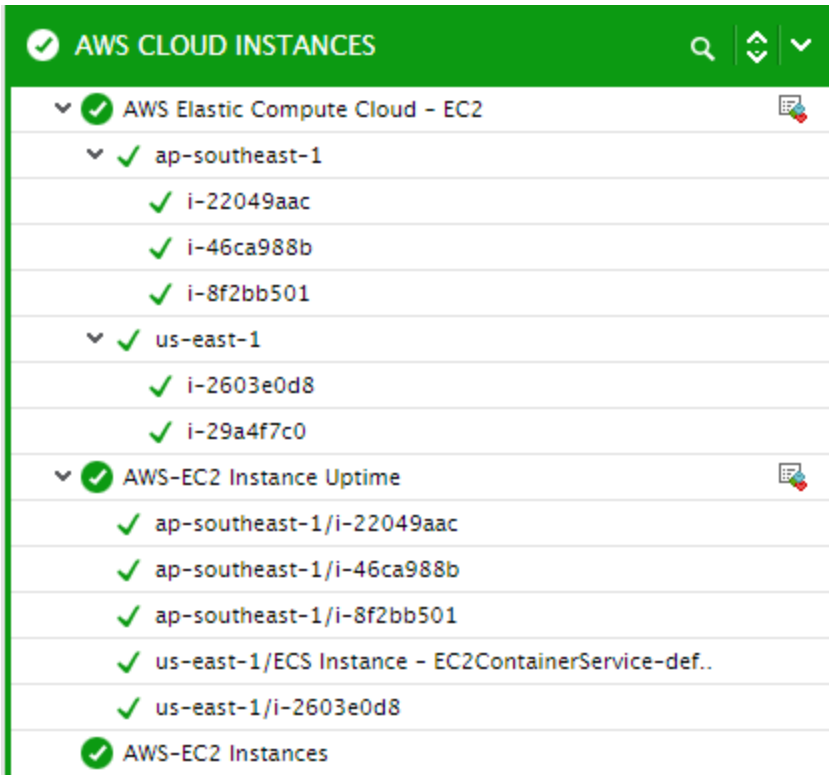


Figure 4.4: The tests mapped to the AWS Cloud Instances layer

4.2.1 AWS-EC2 Instances Test

An Amazon Machine Image (AMI) contains all information necessary to boot instances of your software. For example, an AMI might contain all the software to act as a web server (e.g., Linux, Apache, and your web site) or it might contain all the software to act as a Hadoop node (e.g., Linux, Hadoop, and a custom application). After an AMI is launched, the resulting running system is called an **instance**. All instances based on the same AMI start out identical and any information on them is lost when the instances are terminated or fail.

Users with valid AWS user accounts can sign into the EC2 cloud to view and use available instances, or purchase and launch new ones. With the help of this test, you can determine the total number of instances that are currently available for the configured AWS user account, the number of instances that were newly purchased/terminated, and the count of powered-off instances.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for the AWS EC2 Cloud being monitored

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured

3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and AWS-EC2 VM Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees**. **For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

10. **EXCLUDE INSTANCE** - **This parameter applies only to AWS-EC2 Instance Connectivity, AWS-EC2 Instance Resources , and AWS-EC2 Instance Uptime tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.
11. **DETAILED DIAGNOSIS**- To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Total instances:	Indicates the total number of instances currently available for the configured AWS user account.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the instances available for use for the configured AWS account, regardless of the current state of the instances.
Instances powered on:	Indicates the total number of instances that are currently powered-on.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the powered-on instances available for use for the configured AWS account.
Instances powered off:	Indicates the total number of instances that are currently powered-off.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the powered-off instances available for the configured AWS account.
Added instances:	Indicates the total number of instances that were newly purchased by the configured AWS user account during the last measurement period.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the instances that were newly purchased and launched by the configured AWS user account.

Measurement	Description	Measurement Unit	Interpretation
Removed instances:	Indicates the total number of instances that were newly terminated by the configured AWS user account during the last measurement period.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the instances that were newly terminated/removed by the configured AWS user account.

4.2.2 AWS Elastic Compute Cloud - EC2 Test

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. An EC2 instance is a virtual server in Amazon's Elastic Compute Cloud (EC2) for running applications on the Amazon Web Services (AWS) infrastructure. Since users may run mission-critical applications on these EC2 instances, high uptime of the EC2 instances is imperative to the uninterrupted functioning of these applications and to ensure 100% user satisfaction with this cloud-based service. AWS administrators therefore, should frequently perform health checks on every instance, measure its load and resource usage, and capture potential failures and resource contentions, well before end-users notice and complain. This is exactly where the AWS Elastic Compute Cloud - EC2 test helps!

This test monitors the powered-on state of each EC2 instance and promptly alerts administrators if any instance has been powered-off inadvertently. Additionally, the test also reveals how each instance uses the CPU, disk, and network resources it is configured with, thus providing early pointers to irregularities in instance sizing, and prompting administrators to make necessary amends. This way, the test makes sure that critical applications are always accessible to end-users and perform at peak capacity.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each instance / auto scaling group / instance type / image ID in each region of the AWS cloud being monitored, depending upon the option chosen from the **EC2 FILTER NAME** drop-down

First-level descriptor: AWS EC2 region name

Second-level descriptor: EC2 instance ID / auto scaling group name / instance type / image ID

Test parameters:

1. **TEST PERIOD** - How often should the test be executed

2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **EXCLUDE REGION** - Here, you can provide a comma-separated list of region names or patterns of region names that you do not want to monitor. For instance, to exclude regions with names that contain 'east' and 'west' from monitoring, your specification should be: **east*,*west**
7. **PROXYHOST** and **PROXY PORT** – In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none** , indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **EC2 FILTER NAME** - By default, this test reports metrics for each instance in the AWS infrastructure. This is why, the EC2 Filter Name flag is set to Instance ID by default. Alternatively, you can configure this test to aggregate metrics across a chosen collection of instances, and report one set of metrics per collection. For this, you just need to pick an instance collection from the EC2 Filter Name drop-down. The options available are as follows:
    - *AutoScalingGroupName*: Your EC2 instances can be organized into Auto Scaling Groups so that they can be treated as a logical unit for the purposes of scaling and management. When you create a

group, you can specify its minimum, maximum, and, desired number of EC2 instances.

If you select the *AutoScalingGroupName* option from the **EC2 FILTER NAME** drop-down, then this test will collect metrics for each instance, aggregate the metrics on the basis of the Auto Scaling Groups to which the instances belong, and report metrics for each group.

- *InstanceType*: Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instance types comprise varying combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your applications.

  If the *InstanceType* option is chosen from the **EC2 FILTER NAME** drop-down, then this test will collect metrics for each instance, aggregate the metrics on the basis of the instance type, and report metrics for each type.

- *ImageId*: Instances are created from Amazon Machine Images (AMI). The machine images are like templates that are configured with an operating system and other software, which determine the user's operating environment.

  If the *ImageId* option is chosen from the **EC2 FILTER NAME** drop-down, then this test will collect metrics for each instance, aggregate the metrics on the basis of the AMI using which the instances were created, and report metrics for each image ID.

11. **EXCLUDE INSTANCE** - **This parameter is applicable only if InstanceId is chosen from the EC2 Filter Name drop-down.**

    In this case, against **EXCLUDE INSTANCE**, you can provide a comma-separated list of instance IDs you do not want the test to monitor.

12. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Instance power-on state:	Indicates the current powered-on state of this instance.		This measure is reported only if InstanceID is the option from the EC2 Filter Name drop-down of this test.

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td></td><td></td><td>The values that this measure can report and their corresponding numeric values are detailed in the table below:
<table>
<tr><th>Measure Value</th><th>Description</th><th>Numeric Value</th></tr>
<tr><td>Running</td><td>When the instance is ready for you, it enters the running state.</td><td>1</td></tr>
<tr><td>Pending</td><td>When you launch an instance, it enters the pending state</td><td>2</td></tr>
<tr><td>Terminated</td><td>When you no longer need an instance, you can terminate it, then it goes to terminated state.</td><td>3</td></tr>
<tr><td>Shutting down</td><td>While terminate the instance, As soon as the status of an instance changes to shutting- down or terminated</td><td>4</td></tr>
<tr><td>Stopping</td><td>When you stop</td><td>5</td></tr>
</table>
</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
			<table><tr><td></td><td>your instance, it enters the stopping state</td><td></td></tr><tr><td>**Stopped**</td><td>After exiting the stopping state, it enters the stopped state</td><td>0</td></tr></table> **Note:** By default, this measure will report the **Measure Value**s listed in the table above to indicate the current powered-on state of an instance. In the graph of this measure however, the same will be represented using the numeric equivalents only.
EBS volumes	Indicates the number of EBS volumes attached to this instance.	Number	This measure is reported only if the InstanceId option is chosen from the EC2 Filter Name drop-down of this test. You can attach an EBS volumes to one of your instances that is in the same Availability Zone as the volume. You can attach multiple volumes to the same instance within the limits specified by your AWS account. Your account has a limit on the number of EBS volumes that you can use, and the total storage available to you. Using the detailed diagnosis of this measure, you can identify the volumes that are attached to this EC2 instance.
CPU credit usage:	Indicates the number of CPU credits consumed by this T2 instance / all T2 instances / all T2 instances	Number	This measure is reported only for individual T2 instances, the T2 instance type, and the image ID using which T2 instances (if any) were created. A CPU Credit provides the performance of a full CPU core for one minute. Traditional Amazon EC2 instance types provide fixed performance,

Measurement	Description	Measurement Unit	Interpretation
	created from this image ID during the last measurement period.		while T2 instances provide a baseline level of CPU performance with the ability to burst above that baseline level. The baseline performance and ability to burst are governed by CPU credits. One CPU credit is equal to one vCPU running at 100% utilization for one minute. Other combinations of vCPUs, utilization, and time are also equal to one CPU credit; for example, one vCPU running at 50% utilization for two minutes or two vCPUs running at 25% utilization for two minutes. Each T2 instance starts with a healthy initial CPU credit balance and then continuously (at a millisecond-level resolution) receives a set rate of CPU credits per hour, depending on instance size. When a T2 instance uses fewer CPU resources than its base performance level allows (such as when it is idle), the unused CPU credits (or the difference between what was earned and what was spent) are stored in the credit balance for up to 24 hours, building CPU credits for bursting. When your T2 instance requires more CPU resources than its base performance level allows, it uses credits from the CPU credit balance to burst up to 100% utilization. The more credits your T2 instance has for CPU resources, the more time it can burst beyond its base performance level when more performance is needed. This implies that ideally, the value of the CPU credit usage measure should be low for an instance and the value of the CPU credit balance for that instance should be high, as that way, an instance is assured of more CPU resources when performance demands increase. By comparing the value of this measure across instances, you can precisely identify the instance that has used up a sizeable portion of its CPU credits.
CPU credit balance:	Indicates the number of CPU credits that have been earned by this T2 instance / all T2 instances / all T2 instances created from this image ID	Number	

Measurement	Description	Measurement Unit	Interpretation
CPU utilization:	Indicates the percentage of allocated EC2 compute units that are currently in use on this instance.	Percent	A value close to 100% indicates excessive usage of CPU by an instance. If the value of this measure is consistently high for an instance, it could indicate that the application running on that instance requires more processing power. In such a case, you may want to allocate more CPU resources to that instance.
Disk read operations:	Indicates the rate at which read operations were performed on all disks available to this instance.	Operations/Sec	Compare the value of this measure across instances to know which instance is too slow in processing read requests.
Disk write operations:	Indicates the rate at which write operations were performed on all disks available to this instance.	Operations/Sec	Compare the value of this measure across instances to know which instance is too slow in processing write requests.
Disk reads:	Indicates the rate at which data was read from all disks available to this instance.	KB/Sec	Compare the value of this measure to identify the instance that is the slowest in responding to read requests.
Disk writes:	Indicates the rate at which data was written to all disks available to this instance.	KB/Sec	Compare the value of this measure to identify the instance that is the slowest in responding to write requests.
Incoming network traffic:	Indicates the rate at which data was received by all network interfaces of this instance.	KB/Sec	Compare the value of these measures across instances to know which instance is consuming too much bandwidth. Then, compare the value of the Incoming network traffic and Outgoing network traffic measures of that instance to determine where bandwidth consumption was

Measurement	Description	Measurement Unit	Interpretation
			more - when receiving data over the network? or when sending data?
Outgoing network traffic:	Indicates the rate at which data was sent by all the network interfaces of this instance.	KB/Sec	
EC2 status check:	Indicates whether a status check (system status check or instance status check) failed for this instance		Amazon EC2 performs automated checks on every running EC2 instance to identify hardware and software issues. These status checks are of two types: system and instance status checks. If either of these status checks fails, then this measure will report the value *Failed*. If none of these status checks fail, then this measure will report the value *Passed*. The values that this measure can report and their corresponding numeric values are listed in the table below: **Measure Value** — **Numeric Value** Failed — 1 Passed — 0 **Note:** By default, this measure reports the **Measure Value**s above to indicate whether a check passed or failed. In the graph of this measure however, the same is indicated using the numeric equivalents only.
EC2 instance status check:	Indicates whether/not this instance passed the EC2 instance status check in the last minute.		Instance status checks monitor the software and network configuration of your individual instance. These checks detect problems that require your involvement to repair. When an instance status check fails, typically you will need to address the problem yourself (for example, by rebooting the instance or by making instance configuration changes). The following are examples of problems that can

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td></td><td></td><td>cause instance status checks to fail:
Failed system status checks
Incorrect networking or startup configuration
Exhausted memory
Corrupted file system
Incompatible kernel
The values that this measure can report and their corresponding numeric values are listed in the table below:<table><tr><th>Measure Value</th><th>Numeric Value</th></tr><tr><td>Failed</td><td>1</td></tr><tr><td>Passed</td><td>0</td></tr></table>Note:
By default, this measure reports the Measure Values above to indicate whether a check passed or failed. In the graph of this measure however, the same is indicated using the numeric equivalents only.</td></tr>
<tr><td>EC2 system status check:</td><td>Indicates whether/not this instance passed the EC2 system status check in the last minute.</td><td>Number</td><td>System status checks monitor the AWS systems required to use your instance to ensure they are working properly. These checks detect problems with your instance that require AWS involvement to repair. When a system status check fails, you can choose to wait for AWS to fix the issue, or you can resolve it yourself (for example, by stopping and starting an instance, or by terminating and replacing an instance).
The following are examples of problems that can cause system status checks to fail:
Loss of network connectivity
Loss of system power
Software issues on the physical host
Hardware issues on the physical host</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
			The values that this measure can report and their corresponding numeric values are listed in the table below: **Measure Value** — **Numeric Value** Failed — 1 Passed — 0 **Note:** By default, this measure reports the **Measure Value**s above to indicate whether a check passed or failed. In the graph of this measure however, the same is indicated using the numeric equivalents only.

Detailed Diagnosis:

Using the detailed diagnosis of the **EBS volumes** measure, you can identify the volumes that are attached to a particular EC2 instance.

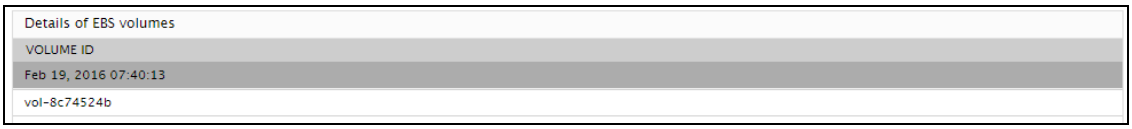


Figure 4.5: The detailed diagnosis of the EBS volumes measure

4.2.3 AWS-EC2 Instance Uptime Test

In cloud-based environments, it is essential to monitor the uptime of server instances launched on the cloud. By tracking the uptime of each of the instances, administrators can determine what percentage of time an instance has been up. Comparing this value with service level targets, administrators can determine the most trouble-prone areas of the infrastructure hosted on the cloud.

In some environments, administrators may schedule periodic reboots of their instances. By knowing that a specific instance has been up for an unusually long time, an administrator may come to know that the scheduled reboot task is not working on an instance.

This test monitors the uptime of each instance available to the configured AWS user account.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each instance launched by the configured AWS user account

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and aws-ec2 vm Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees**. **For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever

does not require authentication by default.

9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **EXCLUDE INSTANCE** - **This parameter applies only to AWS-EC2 Instance Connectivity, AWS-EC2 Instance Resources , and AWS-EC2 Instance Uptime tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.
11. **REPORT MANAGER TIME** - By default, this flag is set to **Yes**, indicating that, by default, the detailed diagnosis of this test, if enabled, will report the shutdown and reboot times of the cloud in the manager's time zone. If this flag is set to **No**, then the shutdown and reboot times are shown in the time zone of the system where the agent is running (i.e., the system system on which the remote agent is running).
12. **DD FREQUENCY** - Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is 1:1. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against DD frequency.
13. **DETAILED DIAGNOSIS**- To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Has the instance been rebooted?:	Indicates whether this instance has been rebooted during the last measurement period or not.	Boolean	If this measure shows 1, it means that the instance was rebooted during the last measurement period. By checking the time periods when this metric changes from 0 to 1, an administrator can determine the times when this instance was rebooted.
Uptime of the	Indicates the time	Secs	If the instance has not been rebooted during

Measurement	Description	Measurement Unit	Interpretation
instance during the last measure period:	period that the instance has been up since the last time this test ran.		the last measurement period and the agent has been running continuously, this value will be equal to the measurement period. If the instance 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 instance 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.
Total uptime of the instance:	Indicates the total time that this instance has been up since its last reboot.	Mins	Administrators may wish to be alerted if an instance has been running without a reboot for a very long period. Setting a threshold for this metric allows administrators to determine such conditions.

4.2.4 AWS-EC2 Instance Resources Test

Tracking the CPU usage, disk and network I/O of every instance launched by a configured AWS user account will provide administrators with valuable insights into how well the instances are utilizing the allocated resources. The **AWS-EC2 Instance Resources** test does just that. This test auto-discovers the instances available for the configured AWS user account, and reports the resource usage of each instance so that, administrators can quickly compare the usage metrics across instances and pinpoint which instance is resource-hungry.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each instance launched by the configured AWS user account

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured

3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and AWS-EC2 VM Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees**. **For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

10. **EXCLUDE INSTANCE** - **This parameter applies only to AWS-EC2 Instance Connectivity, AWS-EC2 Instance Resources , and AWS-EC2 Instance Uptime tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
CPU utilization:	Indicates the percentage of allocated CPU consumed by this instance.	Percent	A high value for this measure indicates that an instance is utilizing CPU excessively - this could be because of one/more resource-intensive processes executing on that instance. Compare the value of this measure across instances to identify the CPU- intensive instances.
Incoming network traffic:	Indicates the rate of incoming network traffic i.e., the rate at which the bytes are received by all the network interfaces connected to this instance.	KB/Sec	Compare the values of these measures across instances to quickly identify the instance that is utilizing the network bandwidth excessively.
Outgoing network traffic:	Indicates the volume of outgoing network traffic i.e., the rate at which the bytes are transferred from all the network interfaces connected to this instance.	KB/Sec	
Disk reads:	Indicates the rate at which data is read from the disks of this instance.	KB/Sec	These measures are good indicators of the level of disk I/O activity on an instance. By comparing the values of these measures across instances, you can accurately determine which instance is performing I/O-intensive operations.
Disk writes:	Indicates the rate at	KB/Sec	

Measurement	Description	Measurement Unit	Interpretation
	which data is written to the disks of this instance.		
Disk read operations:	Indicates the rate at which disk read operations are performed on this instance.	Operations/Sec	These measures are good indicators of the level of disk I/O activity on an instance type. By comparing the values of these measures across types, you can accurately determine the type of instances that is performing I/O-intensive operations.
Disk write operations:	Indicates the rate at which disk write operations were performed on this instance.	Operations/Sec	

4.2.5 AWS-EC2 Aggregated Resource Usage Test

When users launch an instance using the AWS management console, they need to specify the instance type. An instance type is a specification that defines the memory, CPU, storage capacity, and hourly cost for an instance. Some instance types are designed for standard applications, whereas others are designed for CPU-intensive applications, or memory-intensive applications, etc. The different instance types offered by the AWS EC2 cloud are as follows:

Type	CPU	Memory	Local Storage	Platform	I/O	Name
Small	1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit)	1.7 GB	160 GB instance storage (150 GB plus 10 GB root partition)	32-bit	Moderate	m1.small
Large	4 EC2 Compute Units (2 virtual cores with 2 EC2 Compute Units each)	7.5 GB	850 GB instance storage (2 x 420 GB plus 10 GB root partition)	64-bit	High	m1.large
Extra Large	8 EC2 Compute Units (4 virtual cores with 2 EC2 Compute Units each)	15 GB	1690 GB instance storage (4 x 420 GB plus 10 GB root partition)	64-bit	High	m1.xlarge
Micro	Up to 2 EC2 Compute Units (for short periodic bursts)	613 MB	None (use Amazon EBS volumes for storage)	32-bit or 64-bit	Low	t1.micro

High-CPU Medium	5 EC2 Compute Units (2 virtual cores with 2.5 EC2 Compute Units each)	1.7 GB	350 GB instance storage (340 GB plus 10 GB root partition)	32-bit	Moderate	c1.medium
High-CPU Extra Large	20 EC2 Compute Units (8 virtual cores with 2.5 EC2 Compute Units each)	7 GB	1690 GB instance storage (4 x 420 GB plus 10 GB root partition)	64-bit	High	c1.xlarge
High-Memory Extra Large	6.5 EC2 Compute Units (2 virtual cores with 3.25 EC2 Compute Units each)	17.1 GB	420 GB instance storage (1 x 420 GB)	64-bit	Moderate	m2.xlarge
High-Memory Double Extra Large	13 EC2 Compute Units (4 virtual cores with 3.25 EC2 Compute Units each)	34.2 GB	850 GB instance storage (1 x 840 GB plus 10 GB root partition)	64-bit	High	m2.2xlarge
High-Memory Quadruple Extra Large	26 EC2 Compute Units (8 virtual cores with 3.25 EC2 Compute Units each)	68.4 GB	1690 GB instance storage (2 x 840 GB plus 10 GB root partition)	64-bit	High	m2.4xlarge
Cluster Compute	33.5 EC2 Compute Units (2 x Intel Xeon X5570, quad-core "Nehalem" architecture)	23 GB	1690 GB instance 64-bit storage (2 x 840 GB plus 10 GB root partition)	64-bit	Very high (10 Gbps Ethernet)	cc1.4xlarge
Cluster GPU	33.5 EC2 Compute Units (2 x Intel Xeon X5570, quad-core "Nehalem" architecture), plus 2 NVIDIA Tesla M2050 "Fermi" GPUs	22 GB (see note after this table)	1690 GB instance 64-bit storage (2 x 840 GB plus 10 GB root partition)	64-bit	Very high (10 Gbps Ethernet)	cg1.4xlarge

By closely monitoring the CPU usage and the network and disk I/O of each instance type, and comparing these metrics across instance types, you can quickly isolate resource-intensive types. Once again, the test will report metrics for only those types of instances that were launched by the AWS user account configured for the test.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each type of instance launched by the configured AWS user account

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured

3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- This flag applies to the AWS-EC2 VM Resource Usage and aws-ec2 vm Aggregate Resource usage tests only. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees. For the fee details, refer to the AWS EC2 web site**.
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

10. **EXCLUDE INSTANCE** - **This parameter applies only to AWS-EC2 Instance Connectivity, AWS-EC2 Instance Resources , and AWS-EC2 Instance Uptime tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
CPU utilization:	Indicates the percentage of allocated CPU consumed by all instances of this type.	Percent	A high value for this measure indicates that one/more instances of a type are utilizing CPU excessively - this could be because of one/more resource- intensive processes executing on the instances. Compare the value of this measure across types to identify the types of instances that are CPU-intensive.
Incoming network traffic:	Indicates the rate of incoming network traffic i.e., the rate at which the bytes are received by all the network interfaces connected to all the instances of this instance type.	KB/Sec	Compare the values of these measures across instance types to quickly identify the types of instances that are utilizing the network bandwidth excessively.
Outgoing network traffic:	Indicates the volume of outgoing network traffic i.e., the rate at which the bytes are transferred from all the network interfaces connected to all the instances of a particular instance type.	KB/Sec	
Disk reads:	Indicates the rate at which data is read from	KB/Sec	These measures are good indicators of the level of disk I/O activity on an instance type.

Measurement	Description	Measurement Unit	Interpretation
	the disks of all instances of this type.		By comparing the values of these measures across types, you can accurately determine the type of instances that is performing I/O-intensive operations.
Disk writes:	Indicates the rate at which data is written to the disks of all instances of this type.	KB/Sec	
Disk read operations:	Indicates the rate at which disk read operations were performed on the disks of all instances of this type.	Operations/Sec	These measures are good indicators of the level of disk I/O activity on an instance type. By comparing the values of these measures across types, you can accurately determine the type of instances that is performing I/O-intensive operations.
Disk write operations:	Indicates the rate at which disk write operations were performed on the disks of all instances of this type.	Operations/Sec	

4.2.6 AWS-EC2 Instance Connectivity Test

Sometimes, an instance could be in a powered-on state, but the failure of the operating system or any fatal error in internal operations of the instance could have rendered the instance inaccessible to users. In order to enable you to promptly detect such 'hidden' anomalies, this test periodically runs a connectivity check on each instance available for the configured AWS user account, and reports whether the instances are accessible over the network or not.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each instance available for the configured AWS user account

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and AWS-EC2 VM Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees**. **For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use,

then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

10. **EXCLUDE INSTANCE** - **This parameter applies only to AWS-EC2 Instance Connectivity, AWS-EC2 Instance Resources , and AWS-EC2 Instance Uptime tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Avg network delay:	Indicates the average delay between transmission of packets to this instance and receipt of the response to the packet at the source.	Secs	An increase in network latency could result from misconfiguration of the router(s) along the path, network congestion, retransmissions at the network, etc.
Min network delay:	The minimum time between transmission of a packet and receipt of the response back.	Secs	A significant increase in the minimum round-trip time is often a sure sign of network congestion.
Packet loss:	Indicates the percentage of packets lost during transmission from source to target and back.	Percent	Packet loss is often caused by network buffer overflows at a network router or by packet corruptions over the network. The detailed diagnosis for this measure provides a listing of routers that are on the path from the external agent to target server, and the delays on each hop. This information can be used to diagnose the hop(s) that could be causing excessive packet loss/delays.
Network availability of Instance:	Indicates whether the network connection to this instance is available or not.	Percent	A value of 100 indicates that the instance is accessible over the network. The value 0 indicates that the instance is inaccessible. Typically, the value 100 corresponds to a Packet loss of 0.

4.3 The AWS Services Layer

The tests mapped to this layer measures the efficiency some of the critical services offered by the AWS cloud. These include the following:

- The AWS EC2 Elastic Block Store (EBS) service
- The AWS EC2 Container service (ECS)
- The AWS Relational Database service (RDS)
- The AWS Simple Email service (SES)
- The AWS Billing and Cost Management service.

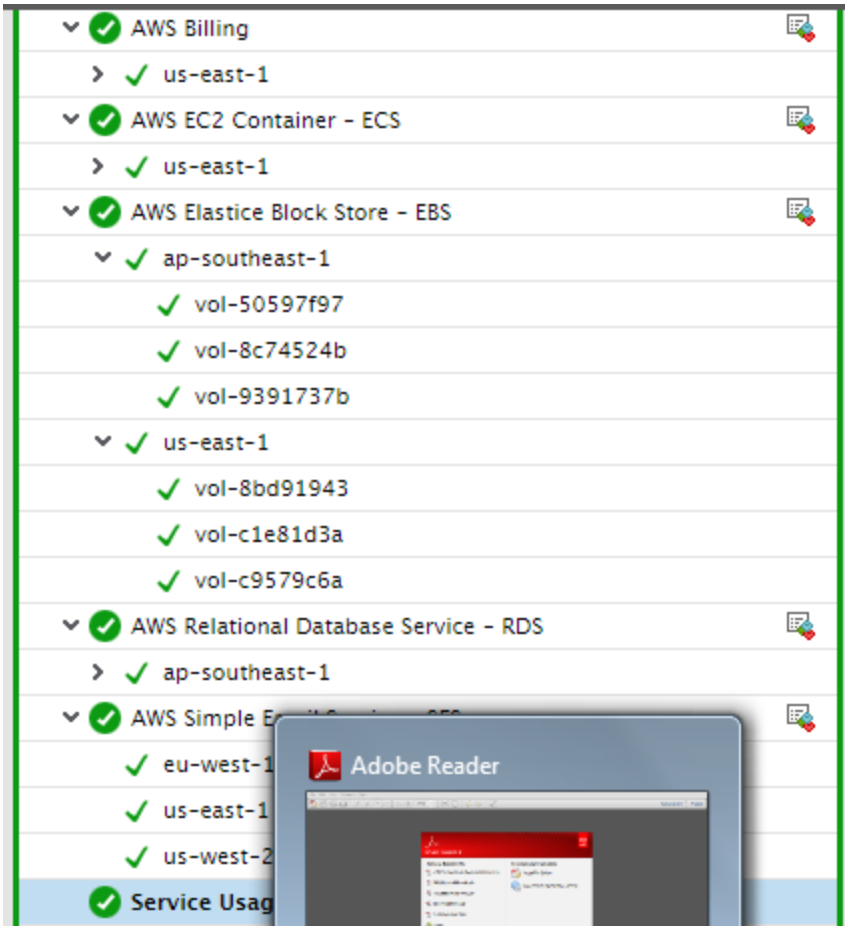


Figure 4.6: The tests mapped to the AWS Services layer

4.3.1 AWS Billing Test

AWS Billing and Cost Management is the service that you use to pay your AWS bill, monitor your usage, and budget your costs.

When budgeting costs, this service also provides forecasts of your estimated costs. Using the AWS Billing test you can configure thresholds for this estimate for each service you subscribe to and also for a roll-up of

estimated charges of all services. The test will then proactively alert you if the estimate is about to exceed your budget, and thus enable you to initiate measures for avoiding cost overruns.

Note:

The metrics of this test can be viewed for the 'us-east' region only. However, since this region stores Amazon CloudWatch metrics for worldwide estimated charges, the *Estimated charges* that this region reports for a service will be the consolidated charges for all regions that are using that particular service.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each service subscribed in each AWS region

First-level descriptor: AWS Region

Second-level descriptor: ServiceName

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **EXCLUDE REGION** - Here, you can provide a comma-separated list of region names or patterns of region names that you do not want to monitor. For instance, to exclude regions with names that contain 'east' and 'west' from monitoring, your specification should be: **east*,*west**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics

collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.

8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Estimated charges:	Indicates the estimated cost of this service across all regions that are using that service.	USD	Compare the value of this measure across services to know which service you will be spending the most on in the future. You can be notified if cost estimations for a service exceed an acceptable limit, by configuring such a limit as a the maximum threshold for this measure for that service. Based on these alarms, you can set out to change how frequently you actually use that service, so as to reduce related overheads. For the Total descriptor, this measure will report the total estimated charges across all services.

4.3.2 AWS EC2 Container - ECS Tests

AWS users can opt to run instances within Elastic Compute Cloud (EC2) or look into using containers. Amazon EC2 Container Service (ECS) manages Docker containers within AWS, allowing users to easily scale up or down and evaluate and monitor CPU usage. These AWS containers run on a managed cluster of EC2 instances, with ECS automating installation and operation of the cluster infrastructure. The first step to get started with ECS therefore is to create a cluster and launch EC2 instances in it. Then, create task definitions. A task is one or more Docker containers running together for one service or a microservice. When

configuring a container in your task definition, you need to define the container name and also indicate how much memory and how many CPU units you want to reserve for each container. Finally, you will have to create a service, so that you can run and maintain a specified number of instances of a task definition simultaneously.

Time and again, administrators will have to check on the resource usage of each cluster, so that they can identify those clusters that have been consistently over-utilizing the CPU and memory resources. Resource usage at the individual service-level should also be monitored, so that administrators can figure out whether the excessive resource consumption by a cluster is because the cluster itself does not have enough resources at its disposal, or because one/more services running on the cluster are depleting the resources. Using the AWS EC2 Container - ECS test, administrators can monitor resource usage both at the cluster and the service-level.

For each AWS region, this test auto-discovers the clusters configured in that region and also the services running on each cluster. CPU and memory usage is then reported for each cluster and service, alongside the CPU and memory reservations (of all tasks) per cluster. These insights help administrators understand where there is a contention for resources - at the cluster-level? or at the service-level? or both? - and accordingly decide what needs to be done to optimize resource usage:

- Should more container instances be added to the cluster to increase the amount of resources at its disposal?
- Should the task definitions of the resource-hungry services be fine-tuned so that the service has more resources to use?

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test:

One set of results for each cluster:service pair in each region of the AWS EC2 cloud

First-level descriptor: AWS EC2 region name

Second-level descriptor: cluster name and/or clustername:servicename

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.

- Once the payment is made, the user will be automatically signed in to the AWS account.
- From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
- Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **EXCLUDE REGION** - Here, you can provide a comma-separated list of region names or patterns of region names that you do not want to monitor. For instance, to exclude regions with names that contain 'east' and 'west' from monitoring, your specification should be: **east*,*west**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none** , indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **ECS FILTER** - By default, this test reports metrics for each cluster and for each service that is running on a cluster. Accordingly, *ServiceName* is the default selection from the **ECS FILTER** drop-down. If you do not want service-level metrics, then you can configure the test to report resource usage at the cluster-level alone. For this, just select *ClusterName* from the **ECS FILTER** drop-down. If this is done, then the test will only report cluster names as descriptors.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
CPU reservation:	The percentage of CPU units that are reserved by running tasks in this cluster.	Percent	**This measure is reported at the cluster- level only - i.e., for the ClusterName descriptor alone.** This value is computed using the following formula:

Measurement	Description	Measurement Unit	Interpretation
			*Total CPU units reserved by ECS tasks on the cluster / Total CPU units that were registered for all the container instances in the cluster * 100* A value close to 100% indicates that almost all resources available to the cluster are being reserved by running tasks in that cluster. This implies that additional services cannot be configured on that cluster until more resources are made available to the cluster or until the CPU reservation of running tasks is reduced.
CPU utilization:	Indicates the percentage of CPU units used by this cluster or by this service	Percent	For a cluster, this value is computed using the following formula: *Total CPU units currently used by ECS tasks on this cluster / Total CPU units that were registered for all the container instances in this cluster * 100* A value close to 100% for this measure at the cluster-level could either indicate that the cluster is resource-starved or that one/more services running on the cluster are consuming excessive resources. If the reason for high CPU usage is the poor resource configuration of the cluster, then, you may want to add more instances to the cluster to add to its resource base. On the other hand, if the cluster is adequately sized with CPU, then you may want to check the value of this measure for each of the services running on the cluster . For a service, this value is computed using the following formula: *Total CPU units currently used by*

Measurement	Description	Measurement Unit	Interpretation
			*ECS tasks defined for this service / Total CPU units that are reserved for the tasks defined for this service * 100* Compare the value of this measure across services of a cluster to know which services of that cluster are guilty of over-utilization of CPU. Once the services are identified, check the CPU reservation of the task definitions of those services to determine whether sufficient resources have been allocated to those tasks. If not, increase the reservations to allow optimal resource usage.
Memory reservation:	The percentage of memory that is reserved by running tasks in this cluster.	Percent	**This measure is reported at the cluster- level only - i.e., for the ClusterName descriptor alone.** This value is computed using the following formula: *Total amount of memory reserved by ECS tasks on the cluster / Total amount of memory that was registered for all the container instances in the cluster * 100* A value close to 100% indicates that almost all resources available to the cluster are being reserved by running tasks in that cluster. This implies that additional services cannot be configured on that cluster until more resources are made available to the cluster or until the memory reservation of running tasks is reduced.
Memory utilization:	Indicates the percentage of memory used by this cluster or by this service	Percent	For a cluster, this value is computed using the following formula: *Total memory currently used by ECS tasks on this cluster / Total memory that is registered for all the container instances in this cluster * 100*

Measurement	Description	Measurement Unit	Interpretation
			A value close to 100% for this measure at the cluster-level could either indicate that the cluster is resource-starved or that one/more services running on the cluster are consuming excessive resources. If the reason for high memory usage is the poor resource configuration of the cluster, then, you may want to add more instances to the cluster to add to its resource base. On the other hand, if the cluster is adequately sized with memory, then you may want to check the value of this measure for each of the services running on the cluster . For a service, this value is computed using the following formula: *Total memory currently used by ECS tasks defined for this service / Total memory reserved for the tasks defined for this service * 100* Compare the value of this measure across services of a cluster to know which services of that cluster are guilty of over-utilization of memory. Once the services are identified, check the memory reservation of the task definitions of those services to determine whether sufficient resources have been allocated to those tasks. If not, increase the reservations to allow optimal resource usage.

4.3.3 AWS Elastic Block Store - EBS Test

Amazon Elastic Block Store (Amazon EBS) provides persistent block level storage volumes for use with Amazon EC2 instances in the AWS Cloud. An Amazon EBS volume is a durable, block-level storage device that you can attach to a single EC2 instance. You can use EBS volumes as primary storage for data that requires frequent updates, such as system drive for an instance or storage for a database application. If such

an EBS volume suddenly becomes unavailable or impaired, it is bound to adversely impact the operations of the EC2 instance attached to that volume, which in turn will damage the experience of the users of that instance. Administrators need to be promptly alerted to such problem conditions, so that they can instantly initiate remedial action and ensure high instance uptime. Besides volume status, administrators also need to track the I/O load on the EBS volume and continuously measure the ability of the volume to handle that load. This insight will enable administrators to provision the volumes with more or less I/O, so as to optimize I/O processing and maximize volume performance. The AWS Elastic Block Store - EBS test helps administrators in this exercise. The test periodically checks the health and availability status of each volume used by the EC2 instances in every region of the AWS EC2 cloud and notifies administrators if any volume is in an abnormal state. Similarly, the test also tracks the I/O load on every volume and measures how well each volume processes the load - overloaded volumes and those that are experiencing processing hiccups are highlighted in the process.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each volume in each region of the AWS cloud being monitored

First-level descriptor: AWS EC2 region name

Second-level descriptor: EBS volume name

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **EXCLUDE REGION** - Here, you can provide a comma-separated list of region names or patterns of region names that you do not want to monitor. For instance, to exclude regions with names that contain

'east' and 'west' from monitoring, your specification should be: **east*,*west**

7. **PROXY HOST AND PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME AND PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none** , indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN AND PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **DD FREQUENCY** - Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is 1:1. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against DD frequency.
11. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
State	Indicates the current state of this volume.		The values that this measure can report and their corresponding numeric values are detailed in the table below: **Measure Value** **Description** **Numeric Value**

			Creating: The volume is being created. The volume will be inaccessible during creation. (0) Available: The volume is available (1) In-use: The volume is in use (2) Deleting: The volume is being deleted (3) Deleted: The volume is deleted (4) Error: Some error has occurred in the volume (5) The detailed diagnosis of this measure will reveal when the volume was created and in which availability zone it resides. **Note:** By default, this measure will report the **Measure Value**s listed in the table above to indicate the current availability state of a volume. In the graph of this measure however, the same will be represented using the numeric equivalents only. If any EBS volume is found to be in an abnormal state, then you can use the detailed diagnosis of this measure to know the volume type, when that volume was created, and in which availability zone the volume resides.
Status	Indicates the current health status of this volume		AWS EC2 periodically runs volume status checks to enable you to better understand, track, and manage potential inconsistencies in the data on an Amazon EBS volume. Volume status checks are automated tests that run every 5 minutes and return a pass or fail status. The value that this measure reports varies with the status reported by the volume

<table>
<tr>
<td></td>
<td></td>
<td></td>
<td>status checks. The table below describes what value this measure reports when , and also lists the numeric values that correspond to the measure values.
<table>
<tr><th>Measure Value</th><th>Description</th><th>Numeric Value</th></tr>
<tr><td>OK</td><td>If all checks pass, the status of the volume is OK.</td><td>0</td></tr>
<tr><td>Impaired</td><td>If a check fails, the status of the volume is impaired</td><td>1</td></tr>
<tr><td>Insufficient-data</td><td>If checks are in progress, then insufficient-data is reported</td><td>2</td></tr>
</table>
Note:
By default, this measure will report the Measure Values listed in the table above to indicate the current status of a volume. In the graph of this measure however, the same will be represented using the numeric equivalents only.</td>
</tr>
<tr>
<td>Idle time:</td>
<td>Indicates the total number of seconds during which no read or write operations were submitted to this volume.</td>
<td>Secs</td>
<td></td>
</tr>
<tr>
<td>Queue length:</td>
<td>Indicates the number of read and write operation requests waiting to be completed.</td>
<td>Number</td>
<td>A consistent increase in the value of this measure could indicate a I/O processing bottleneck on the volume.</td>
</tr>
</table>

Read operations:	Indicates the rate at which read operations were performed on this volume.	Operations/Sec	Compare the value of this measure across volumes to know which volume is too slow in processing read requests.
Write operations:	Indicates the rate at which write operations were performed on this volume.	Operations/Sec	Compare the value of this measure across volumes to know which volume is too slow in processing write requests.
Reads:	Indicates the rate at which data was read from this volume.	KB/Sec	Compare the value of this measure to identify the volume that is the slowest in responding to read requests.
Writes:	Indicates the rate at which data was written to this volume.	KB/Sec	Compare the value of this measure to identify the volume that is the slowest in responding to write requests.
Total read time:	Indicates the total time taken by all completed read operations.	Secs	A very high value for this measure could indicate that the volume took too long to service one/more read requests.
Total write time:	Indicates the total time taken by all completed write operations.	Secs	A very high value for this measure could indicate that the volume took too long to service one/more write requests.
Provisioned IOPS (SSD)volume throughput:	Indicates the percentage of I/O operations per second (IOPS) delivered of the total IOPS provisioned for this volume.	Percent	**This measure will be reported for Provisioned IOPS volumes only.** Provisioned IOPS (SSD) volumes are designed to meet the needs of I/O-intensive workloads, particularly database workloads, that are sensitive to storage performance and consistency in random access I/O throughput. You specify an IOPS rate when you create the volume, and Amazon EBS delivers within 10 percent of the provisioned IOPS performance 99.9 percent of the time over a given year. A Provisioned IOPS (SSD) volume can range in

			size from 4 GiB to 16 TiB and you can provision up to 20,000 IOPS per volume. The ratio of IOPS provisioned to the volume size requested can be a maximum of 30; for example, a volume with 3,000 IOPS must be at least 100 GiB. You can stripe multiple volumes together in a RAID configuration for larger size and greater performance. For smaller I/O operations, you may even see an IOPS value that is higher than what you have provisioned - i.e., the value of this measure can be greater than 100%. This could be because the client operating system may be coalescing multiple smaller I/O operations into a smaller number of large chunks. On the other hand, if the value of this measure is consistently lower than the expected IOPS or throughput you have provisioned, then ensure that your EC2 bandwidth is not the limiting factor; your instance should be EBS-optimized (or include 10 Gigabit network connectivity) and your instance type EBS dedicated bandwidth should exceed the I/O throughput you intend to drive. Another possible cause for not experiencing the expected IOPS is that you are not driving enough I/O to the EBS volumes.
Total IOPS for provisioned IOPS volume:	Indicates the total amount of read and write operations (normalized to 256K capacity units) consumed by this volume in a specified period of time.	Number	**This measure will be reported for Provisioned IOPS volumes only.** I/O operations that are smaller than 256K each count as 1 consumed IOPS. I/O operations that are larger than 256K are counted in 256K capacity units. For example, a 1024K I/O would count as 4 consumed IOPS.
Size:	Indicates the current size of this volume.	GB	For a General Purpose (SSD) Volume, volume size is what dictates the baseline performance level of the volume and how quickly it accumulates I/O credits; larger volumes have

			higher baseline performance levels and accumulate I/O credits faster. For a Provisioned IOPS (SSD) Volume, the ratio of IOPS provisioned to volume size can be a maximum of 30; for example, a volume with 3,000 IOPS must be at least 100 GiB. Magnetic volumes can range in size from 1 GiB to 1 TiB.
Total IOPS:	Indicates the total number of I/O operations that were performed on this volume per second.	Operations/Sec	IOPS are input/output operations per second. Amazon EBS measures each I/O operation per second (that is 256 KiB or smaller) as one IOPS. I/O operations that are larger than 256 KiB are counted in 256 KiB capacity units. For example, a single 1,024 KiB I/O operation would count as 4 IOPS; however, 1,024 I/O operations at 1 KiB each would count as 1,024 IOPS. When you create a 3,000 IOPS volume, either a 3,000 IOPS Provisioned IOPS (SSD) volume or a 1,000 GiB General Purpose (SSD) volume, and attach it to an EBS-optimized instance that can provide the necessary bandwidth, you can transfer up to 3,000 chunks of data per second (provided that the I/O does not exceed the per volume throughput limit of the volume). If your I/O chunks are very large, then the value of this measure may be lesser than what you provisioned because you are hitting the throughput limit of the volume. For example 1,000 GiB General Purpose (SSD) volume has an IOPS limit of 3,000 and a volume throughput limit of 160 MiB/s. If you are using a 256 KiB I/O size, your volume will reach its throughput limit at 640 IOPS (640 x 256 KiB = 160 MiB). For smaller I/O sizes (such as 16 KiB), this same volume can sustain 3,000 IOPS because the throughput is well below 160 MiB/s. On Provisioned IOPS Volumes, for smaller I/O operations, you may even see that the value of

			this measure is higher than what you have provisioned. This could be because the client operating system may be coalescing multiple smaller I/O operations into a smaller number of large chunks. On the other hand, if the value of this measure is consistently lower than the expected IOPS or throughput you have provisioned for a Provisioned IOPS volume, then ensure that your EC2 bandwidth is not the limiting factor; your instance should be EBS-optimized (or include 10 Gigabit network connectivity) and your instance type EBS dedicated bandwidth should exceed the I/O throughput you intend to drive. Another possible cause for not experiencing the expected IOPS is that you are not driving enough I/O to the EBS volumes. Magnetic volumes deliver approximately 100 IOPS on average, with burst capability of up to hundreds of IOPS.
IOPS limits:	Indicates the IOPS limit of this volume.	Operations/Sec	For Provisioned IOPS volumes, the IOPS limit is specified when creating the volumes. For General Purpose IOPS volumes, the volume size dictates the baseline IOPS limit of that volume and how quickly it accumulates I/O credits.
IOPS utilization:	Indicates the percentage of provisioned IOPS or IOPS limit that is being utilized by this volume.	Percent	This metric can also help you identify over-utilized volumes, which could be impacting application performance. In these cases, you could improve performance by upgrading to a different volume type or provisioning more IOPS.
Throughput:	Indicates the rate of reads and writes processed by this volume.	KB/Second	A consistent drop in this value could indicate a I/O processing bottleneck on the volume. You may want to closely track the variations to this measure, so that you can proactively identify the volume that may soon reach its

			throughput limit. The maximum throughput of each volume type is indicated below: • General purpose volumes - 160 MiB/sec • Provisioned IOPS volumes - 320 MiB/sec • Magnetic volumes - 40-90 MiB/sec If your I/O chunks are very large, then a volume will reach its throughput limit much before its IOPS limit is reached. If you are not experiencing the throughput you have provisioned, ensure that your EC2 bandwidth is not the limiting factor; your instance should be EBS-optimized (or include 10 Gigabit network connectivity) and your instance type EBS dedicated bandwidth should exceed the I/O throughput you intend to drive.

Detailed Diagnosis:

The detailed diagnosis of the **State** measure of a volume will reveal when the volume was created and in which availability zone it resides.

Details of Volume

VOLUME TYPE	VOLUME CREATE TIME	VOLUME AVAILABILITY ZONE
Jan 11, 2016 17:09:47		
gp2	Mon Dec 14 19:52:33 IST 2015	ap-southeast-1a

Figure 4.7: The detailed diagnosis of the State measure of the AWS Elastic Block Store - EBS Test

4.3.4 AWS RedShift Test

Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud. The first step to create such a data warehouse is to launch an Amazon Redshift cluster. An Amazon Redshift cluster is a collection of computing resources called nodes. Each cluster runs an Amazon Redshift engine and contains one or more databases. Each cluster has a leader node and one or more compute nodes. The leader node receives queries from client applications, parses the queries, and develops query execution plans. The leader node then coordinates the parallel execution of these plans with the compute nodes, aggregates the intermediate results from these nodes, and finally returns the results back to the client applications. Compute nodes execute the query execution plans and transmit data among themselves to serve these queries. The intermediate results are sent back to the leader node for aggregation before being sent back to the client applications.

Where RedShift is in use, query performance, and consequently, the performance of the dependent client applications, depends upon the following factors:

- Cluster availability
- How the cluster and its nodes use the CPU, network, and storage resources of the cluster;
- Responsiveness of the nodes in the cluster to I/O requests from client applications

To be able to accurately assess whether cluster performance is at the desired level or not, an administrator would require real-time insights into each of the factors listed above. The AWS RedShift test provides administrators with these valuable insights. By reporting the current health status of each cluster managed by RedShift, this test brings unavailable clusters to light. The resource usage of the cluster is also reported, so that potential resource contentions can be proactively isolated. Optionally, you can also configure this test to report metrics for individual nodes in the cluster as well. If this is done, then administrators will be able to instantly drill-down from a resource-hungry cluster to the exact node in the cluster that could hogging the resources. At the node-level, the latency and throughput of each node is also revealed. This way, when users complain of degradation in the performance of client applications, you can quickly identify the cluster and the precise node in the cluster that is slowing down I/O processing and consequently, impacting application performance.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each cluster and/or node in every AWS region on the cloud monitored

First level descriptor: AWS Region

Second level descriptor: Cluster

Third-level descriptor: Node

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".

- Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **EXCLUDE REGION** - Here, you can provide a comma-separated list of region names or patterns of region names that you do not want to monitor. For instance, to exclude regions with names that contain 'east' and 'west' from monitoring, your specification should be: **east*,*west**
7. **REDSHIFT FILTER NAME** - By default, this test reports metrics only for each cluster in each AWS region on the cloud. This is why, this flag is set to **ClusterIdentifier**, by default. If needed, you can configure the test to additionally report metrics for every node in every cluster. For node-level metrics, select the **NodeIdentifer** option from this drop-down. Upon selection, you will be able to view metrics both at the cluster-level and the node-level.
8. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
9. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
10. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
CPU utilization:	Indicates the percentage of CPU utilized by this cluster/node.	Percent	For a cluster, this measure will report the aggregate CPU usage of all nodes in the cluster. If the value of this measure is consistently above 50% for a cluster, it indicates that a serious resource contention may occur on that cluster, if additional processing power is not provided to it. In such a case, you may want to consider adding more nodes to the cluster, or adding more CPUs to

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td></td><td></td><td>the existing nodes.

You can also compare the CPU usage of nodes in the resource-hungry cluster to determine whether one/more nodes are hogging the CPU. If so, you may want to tweak the load-balancing algorithm of your cluster to ensure uniform load distribution.</td></tr>
<tr><td>Database connections:</td><td>Indicates the number of connections to the databases in this cluster.</td><td>Number</td><td>This measure is only reported at the cluster-level and not the node-level.</td></tr>
<tr><td>Health status:</td><td>Indicates the current health status of this cluster.</td><td>Percent</td><td>This measure is only reported at the cluster-level and not the node-level.

Every minute the cluster connects to its database and performs a simple query. If it is able to perform this operation successfully, then the value of this measure will be Healthy. Otherwise, the value of this measure will be Unhealthy. An Unhealthy status can occur when the cluster database is under extremely heavy load or if there is a configuration problem with a database on the cluster.

The numeric values that correspond to the measure values mentioned above are as follows:

<table><tr><th>Measure Value</th><th>Numeric Value</th></tr><tr><td>Healthy</td><td>1</td></tr><tr><td>Unhealthy</td><td>0</td></tr></table>
Note:

This measure will report one of the Measure Values listed above to indicate the current state of a cluster. In the graph of this measure however, cluster status will be indicated using</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
			the numeric equivalents only.
Is maintenance mode?:	Indicates whether/not this cluster is in the maintenance mode presently.		The values that this measure can report and their corresponding numeric values are listed in the table below: **Measure Value** — **Numeric Value** Yes — 1 No — 0 **Note:** • This measure will report one of the **Measure Value**s listed above to indicate whether/not a cluster is in the maintenance mode. In the graph of this measure however, the same will be indicated using the numeric equivalents only. • This measure is reported only at the cluster-level and not the node-level. • Even though your cluster might be unavailable due to maintenance tasks, the *Health status* measure of the test will report the value *Healthy* for that cluster
Network receive throughput:	Indicates the rate at which this cluster or node receives data.	KB/Secs	For a cluster, a consistent increase in the value of these measures is indicative of excessive usage of network resources by the cluster. In such a case, compare the value of these measures across the nodes of a cluster to identify the nodes that are over- utilizing network bandwidth.
Network transmit throughput:	Indicates the rate at which this cluster or node sends data.	KB/Secs	
Disk space used:	Indicates the percentage of disk space used by this cluster/node.	Percent	If the value of this measure is close to 100% for a cluster, it indicates that the cluster is rapidly running out of storage resources. You may want to consider adding more nodes to

Measurement	Description	Measurement Unit	Interpretation
			the cluster to increase the storage space available. Alternatively, you can add fewer nodes and yet significantly increase the cluster resources by opting for node types that are by default large-sized and hence come bundled with considerable storage space. When a cluster's storage resources are rapidly depleting, you may want to compare the space usage of the nodes in cluster, so that you can quickly isolate that node that is eroding the space. Tweaking your load-balancing algorithm could go a long way in eliminating such node overloads.
Read IOPS:	Indicates the average number of disk read operations performed by this node per second.	Reads/Sec	A high value is desired for this measure, as that's the trait of a healthy node. You can compare the value of this measure across nodes to identify the node that is slowest in processing read requests.
Read latency:	Indicates the average amount of time taken by this node for disk read I/O operations.	Reads/Sec	Ideally, the value of this measure should be very low. Its good practice to compare the value of this measure across nodes of a cluster and isolate those nodes in the cluster where the value of this measure is abnormally high. Such nodes slow down I/O processing and adversely affect application performance.
Read throughput:	Indicates the average number of bytes read from disk by this node per second.	KB/Sec	A high throughput signifies faster processing of read I/O requests. A low throughput is indicative of slow read request processing. Compare the value of this measure across nodes of a cluster to isolate those nodes that have registered an abnormally low value for this measure. Such nodes not only affect cluster performance, but also the performance of dependent client applications.

Measurement	Description	Measurement Unit	Interpretation
Write IOPS:	Indicates the average number o disk write operations performed by this node per second.	Writes/Sec	A high value is desired for this measure, as that's the trait of a healthy node. You can compare the value of this measure across nodes to identify the node that is slowest in processing write requests.
Write latency:	Indicates the average amount of time taken by this node for disk write I/O operations.	Secs	Ideally, the value of this measure should be very low. Its good practice to compare the value of this measure across nodes of a cluster and isolate those nodes in the cluster where the value of this measure is abnormally high. Such nodes slow down I/O processing and adversely affect application performance.
Write throughput:	Indicates the average number of bytes written to disk by this node per second.	KB/Sec	A high throughput signifies faster processing of write I/O requests. A low throughput is indicative of slow write request processing. Compare the value of this measure across nodes of a cluster to isolate those nodes that have registered an abnormally low value for this measure. Such nodes not only affect cluster performance, but also the performance of dependent client applications.

4.3.5 AWS Relational Database Service - RDS Test

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizeable capacity for an industry-standard relational database and manages common database administration tasks. It also manages backups, software patching, automatic failure detection, and recovery.

The basic building block of Amazon RDS is the DB instance. A DB instance is an isolated database environment in the cloud. A DB instance can contain multiple user-created databases, and you can access it by using the same tools and applications that you use with a stand-alone database instance.

Each DB instance runs a DB engine. Amazon RDS currently supports the MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server DB engines.

The computation and memory capacity of a DB instance is determined by its DB instance class. For each DB instance, you can select from 5 GB to 6 TB of associated storage capacity. Each DB instance class has minimum and maximum storage requirements for the DB instances that are created from it. You can select

the DB instance that best meets your needs. If your needs change over time, you can change DB instances. For example, initially, you may have launched a standard (previous generation) DB instance, which provides a balance of compute, memory, and network resources for your applications. However later, based on usage, you may have realized that a burst capable - current generation DB instance, with the capability to burst to full CPU usage, is ideal for your needs. In such circumstances, RDS facilitates the switch from one type DB instance to another. But to understand which DB instance class best suits your needs and make timely and accurate adjustments to your DB instance class selection, you will have to constantly track the CPU, memory, network, and space usage of each active DB instance on the cloud and derive usage patterns. Also, to ensure optimal storage performance, you additionally need to keep an eye on the I/O operations performed on the DB instances and identify latent DB instances. This is exactly what the AWS Relational Database Service - RDS enables you to achieve.

This test closely tracks the current status, resource usage, and I/O activity of every active DB instance on each cloud region, and brings the following to light:

- Is any DB instance in an abnormal state presently?
- How are the DB instances using the CPU resources they have been configured with? Is any DB instance consuming high levels of CPU consistently? Should the DB instance class be changed?
- Does the DB instance have enough RAM? Will changing the DB instance class help in reducing the memory pressure on the instance?
- Do any db.t2 instances have a poor CPU credit balance?
- Is the disk I/O queue of any DB instance abnormally high? Which instance is this and when is I/O latency on that instance very high - when reading from or writing to the instance?
- Which DB instance is hungry for network bandwidth?
- Do all DB instances have enough free space? If not, which ones are rapidly running short of space?

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each active DB instance in each region of the AWS EC2 cloud

First-level descriptor: AWS EC2 region name

Second-level descriptor: DB instance identifier / DB instance class / DB engine name, depending upon the option you choose from the **RDS FILTER** drop-down

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.

- Provide the details of the user for whom you wish to create the AWS account.
- Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
- Once the payment is made, the user will be automatically signed in to the AWS account.
- From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
- Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **EXCLUDE REGION** - Here, you can provide a comma-separated list of region names or patterns of region names that you do not want to monitor. For instance, to exclude regions with names that contain 'east' and 'west' from monitoring, your specification should be: **east*,*west**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **RDS FILTER** - By default, this test reports metrics for each active DB instance on the cloud. This is why, this flag is set to **DBInstanceIdentifier**, by default. If needed, you can pick either of the following options from this drop-down:
    - **DatabaseClass:** The computation and memory capacity of a DB instance is determined by its DB instance class. If you select this option, then this test will report metrics for each DB instance class. In other words, eG will aggregate metrics for all databases that belong to a DB intance class, and will present these metrics at the macro class-level.
    - **EngineName:** Each DB instance runs a DB engine. Amazon RDS currently supports the MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server DB engines. Each DB engine has its own supported features, and each version of a DB engine may include specific features. If you select this option, then this test will report metrics for every DB engine. In this case, eG will aggregate metrics for all databases using a particular engine, and will present these metrics at the macro engine-level.

Measures reported by the test:

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td>RDS instance status:</td><td>Indicates the current status of this DB instance.</td><td></td><td>This measure is reported only for a DB instance descriptor.

The values that this measure reports, the significance of each of these values, and the numeric values that correspond to them are discussed in the table below:

<table>
<tr><th>Measure Value</th><th>Description</th><th>Numeric Value</th></tr>
<tr><td>Failed</td><td>The instance has failed and Amazon RDS was unable to recover it. Perform a point- in- time restore to the latest restorable time of the instance to recover the data.</td><td>0</td></tr>
<tr><td>Available</td><td>The instance is healthy and available</td><td>1</td></tr>
<tr><td>Backing-up</td><td>The instance is currently being backed up.</td><td>2</td></tr>
<tr><td>Creating</td><td>The instance is being created. The instance is inaccessible while it is being created.</td><td>3</td></tr>
</table>
</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
			Inaccessible-encryption-credentials: The KMS key used to encrypt or decrypt the DB instance could not be accessed. 4 Incompatible-credentials: The supplied CloudHSM username or password is incorrect. Please update the CloudHSM credentials for the DB instance. 5 Incompatible-network: Amazon RDS is attempting to perform a recovery action on an instance but is unable to do so because the VPC is in a state that is preventing the action from being completed. This status can occur if, for example, all available IP addresses in a subnet were in 6

Measurement	Description	Measurement Unit	Interpretation		
				use and Amazon RDS was unable to get an IP address for the DB instance.	
			Incompatible-option-group	Amazon RDS attempted to apply an option group change but was unable to do so, and Amazon RDS was unable to roll back to the previous option group state. Consult the Recent Events list for the DB instance for more information. This status can occur if, for example, the option group contains an option such as TDE and the DB instance does not contain encrypted information.	7

Measurement	Description	Measurement Unit	Interpretation
			Incompatible-parameters: Amazon RDS was unable to start up the DB instance because the parameters specified in the instance's DB parameter group were not compatible. Revert the parameter changes or make them compatible with the instance to regain access to your instance. Consult the Recent Events list for the DB instance for more information about the incompatible parameters. — 8 Incompatible-restore: Amazon RDS is unable to do a point-in-time restore. Common causes for this status include — 9

Measurement	Description	Measurement Unit	Interpretation
			using temp tables or using MyISAM tables.
			Maintenance: Amazon RDS is applying a maintenance update to the DB instance. — 10
			Modifying: The instance is being modified because of a customer request to modify the instance. — 11
			Rebooting: The instance is being rebooted because of a customer request or an Amazon RDS process that requires the rebooting of the instance. — 12
			Renaming: The instance is being renamed because of a customer request to rename it. — 13
			Resetting-master-credentials: The master credentials for the instance — 14

Measurement	Description	Measurement Unit	Interpretation

	are being reset because of a customer request to reset them.	
Restore-error	The DB instance encountered an error attempting to restore to a point- in- time or from a snapshot.	15
Upgrading	The database engine version is being upgraded.	16
Storage-full	The instance has reached its storage capacity allocation. This is a critical status and should be remedied immediately; you should scale up your storage by modifying the DB instance. Set alarms to warn you when storage space is getting low so	17

Measurement	Description	Measurement Unit	Interpretation
			<table><tr><td></td><td>you don't run into this situation.</td><td></td></tr><tr><td>Deleting</td><td>The instance is being deleted.</td><td>18</td></tr></table> **Note:** This measure reports the Measure Values listed in the table above to indicate the current status of a DB instance. In the graph of this measure however, the same will be represented using the corresponding numeric equivalents only.
CPU credit usage:	Indicates the number of CPU units consumed by this T2 DB instance/ all DB instances that belong to this T2 DB instance class / all T2 DB instances using this DB engine, during the last measurement period.	Number	These measures are reported only for individual T2 instances, instances that belong to T2 DB instance classes, and the DB engines used only by T2 instances. A CPU Credit provides the performance of a full CPU core for one minute. Traditionalinstance types provide fixed performance, while T2 instances provide a baseline level of CPU performance with the ability to burst above that baseline level. The baseline performance and ability to burst are governed by CPU credits. One CPU credit is equal to one vCPU running at 100% utilization for one minute. Other combinations of vCPUs, utilization, and time are also equal to one CPU credit; for example, one vCPU running at 50% utilization for two minutes or two vCPUs running at 25% utilization for two minutes. Each T2 instance starts with a healthy initial CPU credit balance and then continuously (at a millisecond-level resolution) receives a set rate of CPU credits per hour, depending on instance size.

Measurement	Description	Measurement Unit	Interpretation
			When a T2 instance uses fewer CPU resources than its base performance level allows (such as when it is idle), the unused CPU credits (or the difference between what was earned and what was spent) are stored in the credit balance for up to 24 hours, building CPU credits for bursting. When your T2 instance requires more CPU resources than its base performance level allows, it uses credits from the CPU credit balance to burst up to 100% utilization. The more credits your T2 instance has for CPU resources, the more time it can burst beyond its base performance level when more performance is needed. This implies that ideally, the value of the CPU credit usage measure should be low for an instance and the value of the CPU credit balance for that instance should be high, as that way, an instance is assured of more CPU resources when performance demands increase. By comparing the value of this measure across instances, you can precisely identify the instance that has used up a sizeable portion of its CPU credits.
CPU credit balance:	Indicates the number of CPU credits that an instance has accumulated.	Number	
CPU utilization:	Indicates the percentage of CPU utilized by this DB instance / DB instance class / DB engine	Percent	A value close to 100% for this measure for any DB instance is indicative of excessive CPU usage by that instance. Track the variations to the value of this measure for such an instance closely, and figure out whether CPU usage is consistently high and close to 100%. If so , you can conclude that the instance requires more CPU than what's been allocated to it. You may want to change to the DB instance class definition to allot more CPU resources to all instances it governs.
Binlog disk usage:	Indicates the amount of disk space occupied by binary	KB	The binary log on MySQL has two important purposes: • For replication, the binary log on a master

Measurement	Description	Measurement Unit	Interpretation
	logs on this DB instance / all DB instances of this DB instance class / all DB instances using this DB engine		replication server provides a record of the data changes to be sent to slave servers. The master server sends the events contained in its binary log to its slaves, which execute those events to make the same data changes that were made on the master. . • Certain data recovery operations require use of the binary log. After a backup has been restored, the events in the binary log that were recorded after the backup was made are re-executed. These events bring databases up to date from the point of the backup. Typically, MySQL uses several logging formats to record information in the binary log. There are three logging formats: • Replication capabilities in MySQL originally were based on propagation of SQL statements from master to slave. This is called statement-based logging. • In row-based logging, the master writes events to the binary log that indicate how individual table rows are affected. • A third option is also available: mixed logging. With mixed logging, statement-based logging is used by default, but the logging mode switches automatically to row-based in certain cases. MySQL on Amazon RDS supports both the row-based and mixed binary logging formats for MySQL version 5.6. The default binary logging format is mixed. For DB instances running MySQL versions 5.1 and 5.5, only mixed binary logging is supported. If the value of this measure grows consistently, it could mean that large binary files are being created. At this juncture, you may want to check the logging format configured for MySQL on

Measurement	Description	Measurement Unit	Interpretation
			Amazon RDS. This is because, very often, row-based binary logging format can result in very large binary log files. If you do not change the logging mode, then such files will continue to be created, thereby reducing the amount of storage space available for a DB instance. This in turn can increase the amount of time to perform a restore operation of a DB instance.
Database connections:	Indicates the number of database connections currently used by this instance / all instances that belong to this DB instance class / all instances using this DB engine	Number	
Disk queue depth:	Indicates the number of outstanding IOs (read/write requests) waiting to access this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine.	Number	If the value of this measure keeps increasing steadily and significantly for a DB instance, it could indicate that the DB instance is latent, and is unable to process I/O requests quickly. The value of this measure therefore should be low at all times.
Freeable memory:	Indicates the amount of available random access memory for this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine	MB	A high value is desired for this measure to ensure peak performance of a DB instance.

Measurement	Description	Measurement Unit	Interpretation
Replica lag time:	Indicates the amount of time a Read Replica DB Instance lags behind this source DB Instance / all source DB instances that belong to this DB instance class / all source DB instances using DB engine	Secs	**This measure applies to MySQL read replicas only.** f your system runs on Amazon Relational Database Service (RDS) you may have opted to configure one or more replicas for your main MySQL database(s). This means you have a master RDS instance and at least one slave RDS instance which receives updates from the master. This process is called replication. Replication ensures that changes made on the master database also happen on the slave after some period of time. For a variety of reasons this period of time can increase. For example, a long-running query or erroneous query can cause replication to slow down or stop entirely. This results in replication lag: changes made on your main database aren't showing up on the slave replica because the replica is lagging behind. If the value of this measure is increasing consistently for a DB instance, it is a cause for concern, as it indicates that the slave is not in sync with the master and will take a long time to catch up. If for any reason the master DB instance fails at this juncture, there is bound to be significant data loss owing to the master-slave non-sync. When there is a replication issue the output of *show slave status*; is quite useful in debugging and resolving it. You need to review the values of: *Slave_SQL_Running* *Last_Error* *Last_SQL_Error* When a particular SQL query failed on the slave it could be that execution of queries in general has stopped. This is indicated by *Slave_ SQL_*

Measurement	Description	Measurement Unit	Interpretation
			Running having the value *No*. In that case you'll either need to: • Remedy the error by fixing the issue that caused the SQL query to fail. • Decide to resume replication by letting the slave ignore that error. The former situation can be tricky as it requires you to figure out what data or query is problematic based on the values of *Last_Error* and *Last_SQL_Error*. These fields may provide enough information to determine any incorrect records but this is not always the case. In the latter case you would execute the following command on the slave: *CALL mysql.rds_skip_repl_error;* You should only run this command when you've determined that skipping the SQL query won't lead to inconsistent data or incorrect data on the slave (or, at least, that this is allowed to occur by skipping that particular SQL query).
Swap usage:	Indicates the amount of swap space used on this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine.	KB	
Read IOPS:	Indicates the rate at which disk read I/O operations were performed by this DB instance / all DB instances that	Reads/Sec	Ideally, the value of this measure should be high. A consistent drop in this value could indicate a read latency.

Measurement	Description	Measurement Unit	Interpretation
	belong to this DB instance class / all DB instances using this DB engine		
Write IOPS:	Indicates the rate at which disk write I/O operations were performed by this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine	Writes/Sec	Ideally, the value of this measure should be high. A consistent drop in this value could indicate a write latency.
Read latency:	Indicates the average amount of time this DB instance / all DB instances of this instance class / all DB instances using this engine, took to service read requests.	Secs	Ideally, the value of this measure should be low. A consistent rise in this value could indicate a read latency. Compare the value of this measure across DB instances to know which instance is the slowest in servicing read requests.
Write latency:	Indicates the average amount of time this DB instance / all DB instances of this instance class / all DB instances using this engine, took to service write requests.	Secs	Ideally, the value of this measure should be low. A consistent rise in this value could indicate a write latency. Compare the value of this measure across DB instances to know which instance is the slowest in servicing write requests.
Read throughput:	Indicates the rate at which data was read from the disk by this	KB/Sec	Ideally, the value of this measure should be high. A steady decrease in this value could indicate a read latency. Compare the value of this measure

Measurement	Description	Measurement Unit	Interpretation
	DB instance / all DB instances of this instance class / all DB instances using this DB engine.		across DB instances to know which instance is the slowest in servicing read requests.
Write throughput:	Indicates the rate at which data was written to the disk by this DB instance / all DB instances of this instance class / all DB instances using this DB engine.	KB/Sec	Ideally, the value of this measure should be high. A steady decrease in this value could indicate a write latency. Compare the value of this measure across DB instances to know which instance is the slowest in servicing write requests.
Network receive throughput:	Indicates the incoming network traffic on this DB instance / all DB instances that belong to this instance class / all DB instances using this engine.	KB/Secs	The value of these measures includes both customer database traffic and Amazon RDS traffic used for monitoring and replication. A high value for these measures is indicative of high bandwidth usage by a DB instance. Under such circumstances, compare the value of the Network receive throughput measure with that of th Network transmit throughput measure to determine when the maximum bandwidth was consumed - when sending data or when receiving it?
Network transmit throughput:	Indicates the outgoing network traffic on this DB instance / all DB instances that belong to this instance class / all DB instances using this engine.	KB/Secs	
Total storage space:	Indicates the total amount of storage space allocated to this DB instance / all	MB	

Measurement	Description	Measurement Unit	Interpretation
	DB instances that belong to this instance class / all DB instances using this DB engine.		
Used storage space:	Indicates the amount of storage space used by this DB instance / all DB instances that belong to this instance class / all DB instances using this DB engine.	MB	Compare the value of this measre across DB instances to know which instance is consuming storage space excessively.
Free storage space:	Indicates the amount of storage space still unused by this DB instance / all DB instances that belong to this instance class / all DB instances using this DB engine.	MB	A high value for this measure is ideal. Compare the value of this measure across DB instances to know which instance is left with very little free space.
Free storage space:	Indicates the percentage of storage space allocated to this DB instance / all DB instances that belong to this instance class / all DB instances using this DB engine, which is still available for use.	Percent	A value close to 100% is desired for this measure. If the value of this measure is below 50% consistently, it indicates that the DB instance is not sized with adequate resources. You may want to consider changing the DB instance class of that instance, so that more storage resources are available to it.

4.3.6 AWS Simple Email Service - SES Test

Amazon Simple Email Service (Amazon SES) is a cost-effective email service built on the reliable and scalable infrastructure that Amazon.com developed to serve its own customer base. This service allows you to build an email functionality into an application that you are running on AWS. With Amazon SES, you can send transactional email, marketing messages, or any other type of high-quality content to your customers. You can also use Amazon SES to receive messages and deliver them to an Amazon S3 bucket, call your custom code via an AWS Lambda function, or publish notifications to Amazon SNS.

Amazon SES has a set of sending limits to regulate the number of email messages that you can send and the rate at which you can send them. Depending upon the level of email activity in your environment, you may want to modify these limits, as any violation will result in mails not being sent at all. You may hence have to closely study the email activity in your environment and determine whether/not the sending limits need to be fine-tuned. The **AWS Simple Email Service - SES** test helps with this! By reporting the send quotas configured along with the count of mails sent and the send rate for each AWS region, this test readily provides you with all the information you need to take the right decision with regards to whether/not the quota needs to be reset.

Also, the key measure of the performance of any email service is successful message delivery. If a majority of the delivery attempts made at any given point in time resulted in bounces, rejections, or complaints, it is a problem condition that warrants an investigation. The **AWS Simple Email Service - SES** test proactively alerts you to such abnormalities! For each region, the test reports the count and percentage of emails bounced, mails rejected, and complaints received, and notifies you if these values exceed acceptable limits.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for each AWS region

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with

the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
7. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
8. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the proxy domain and proxy workstation parameters. If the environment does not support a Windows NTLM proxy, set these parameters to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Sending quota:	Indicates the maximum number of emails that can be sent by this region in a day.	Emails/Day	The sending quota reflects a rolling time period. Every time you try to send an email, Amazon SES checks how many emails you sent in the previous 24 hours. As long as the total number of emails that you have sent is less than your quota, your send request will be accepted and your email will be sent. If you have already sent your full quota, your send request will be rejected with a throttling exception. You will not be able to send more emails until some of the previous sending rolls out of its 24-hour window.
Total sent:	Indicates the total number of emails that this region sent during the last 24 hours.	Number	If the value of this measure keeps growing closer to the value of the value of the Sending quota measure, it implies a high level of email activity in the region. Under such

Measurement	Description	Measurement Unit	Interpretation
			circumstances, it is best to increase the sending quota, so that the quota is not violated, causing SES to stop sending emails.
Current sents:	Indicates the number of emails that this region sent during the last measurement period.	Number	
Sents:	Indicates the percentage of sending quota that this region exhausted in the last 24 hours.	Percent	This measure is computed using the following formulat: (Total sent / Sending quota) * 100 If the value of this measure is consistently higher than 50%, it implies a high level of email activity in the region. Under such circumstances, it is best to increase the sending quota, so that the quota is not violated, causing SES to stop sending emails.
Max send rate:	Indicates the maximum number of emails that this region can send per second.	Emails/Sec	You can exceed this limit for short bursts, but not for a sustained period of time.
Total bounces:	Indicates the total number of emails bounced to this region during the last 24 hours.	Number	An email is hard-bounced when the email is rejected by the recipient's ISP or rejected by Amazon SES because the email address is on the Amazon SES suppression list. This measure reports the count of hard bounces alone.
Current bounces:	Indicates the number of emails that were bounced to this region during the last measurement period.	Number	The value of this measure should be kept at a minimum, as excessive bounces constitute abuse and can put your AWS account at the risk of termination.
Bounce:	Indicates the	Percent	Ideally, the value of this measure should be

Measurement	Description	Measurement Unit	Interpretation
	percentage of emails that were bounced to this region during the last measure period.		very low. A high value constitutes abuse and can put your AWS account at the risk of termination.
Complaints:	Indicates the number of complaints received by this region during the last measure period.	Number	If an email is accepted by the ISP and delivered to the recipient, but the recipient does not want the email and clicks a button such as "Mark as spam.", then SES will send you a complaint notification. The value of this measure should be kept at a minimum, as a large number of complaints constitute abuse and can put your AWS account at the risk of termination.
Complaint:	Indicates the percentage of complaints received by this region during the last measure period.	Percent	Ideally, the value of this measure should be very low. A high value constitutes abuse and can put your AWS account at the risk of termination.
Total rejected:	Indicates the number of emails sent by this region that were rejected during the last 24 hours.	Number	A rejected email is an email that Amazon SES initially accepted, but later rejected because the email contained a virus. Amazon SES notifies you by email and does not send the message. A high value for this measure is a cause for concern as it could indicate that your email system is severely infected.
Current rejected:	Indicates the number of emails sent by this region that were rejected during the last measurement period.	Number	A consistent rise in the value for this measure is a cause for concern as it could indicate that your email system is severely infected.
Rejected:	Indicates the percentage of emails	Percent	A high value for this measure is a cause for concern as it could indicate that your email

Measurement	Description	Measurement Unit	Interpretation
	sent by this region that were rejected during the last measurement period.		system is severely infected.
Total delivery attempts:	Indicates the number of emails that were successfully delivered by this region to the recipient's mail server during the last 24 hours.	Number	A high value is desired for this measure.
Current delivery attempts:	Indicates the number of emails that were successfully delivered by this region to the recipient's mail server during the last measurement period.	Number	A consistent drop in the value of this measure is a cause of concern.

4.3.7 Service Usage Test

Use this test to receive an overview of the instances launched and services (EBS and RDS) used by a monitored AWS user account. Understanding how many instances of service are utilized by an account will help you to bill that user accordingly.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agent

Output of the test: One set of results for the AWS account configured for monitoring

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:

- Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
- Provide the details of the user for whom you wish to create the AWS account.
- Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
- Once the payment is made, the user will be automatically signed in to the AWS account.
- From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
- Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
7. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
8. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
9. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
EC2 instances:	Indicates the total number of EC2 instances currently available for the configured AWS user account.	Number	Use the detailed diagnosis of this measure to know which are the available instances.
EC2 instances poweredon:	Indicates the total number of instances that are currently powered- on for the configured AWS user account.	Number	Use the detailed diagnosis of this measure to know the names of the powered-on instances.
EBS volumes:	Indicates the total number of EBS volumes currently available for the configured AWS user account.	Number	Use the detailed diagnosis of this measure to know which volumes are available for use presently.
RDS instances:	Indicates the total number of RDS instances that are configured for the AWS user account.	Number	Use the detailed diagnosis of this measure to know the details of all RDS instances configured for the AWS user account.
RDS instances available:	Indicates the number of instances that are currently powered- on and available for the use of the configured AWS user account.	Number	Use the detailed diagnosis of this measure to know the details of all RDS instances that are powered-on and available for the use of the AWS user account.

The detailed diagnosis of the **EC2 instances** measure lists the names of all EC2 instances available for the configured AWS user account.

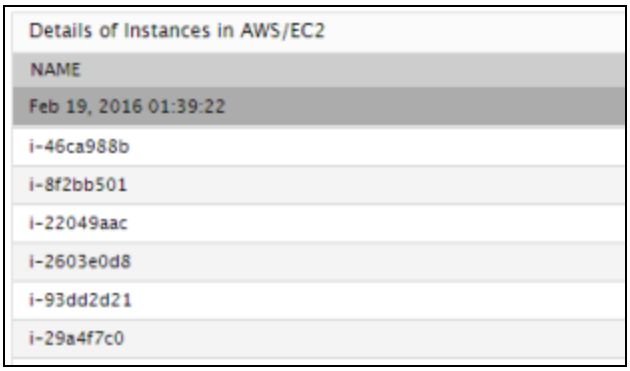


Figure 4.8: The detailed diagnosis of the EC2 instances measure

The detailed diagnosis of the **EC2 instances poweredon** measure lists the names of the powered-on instances.alone.

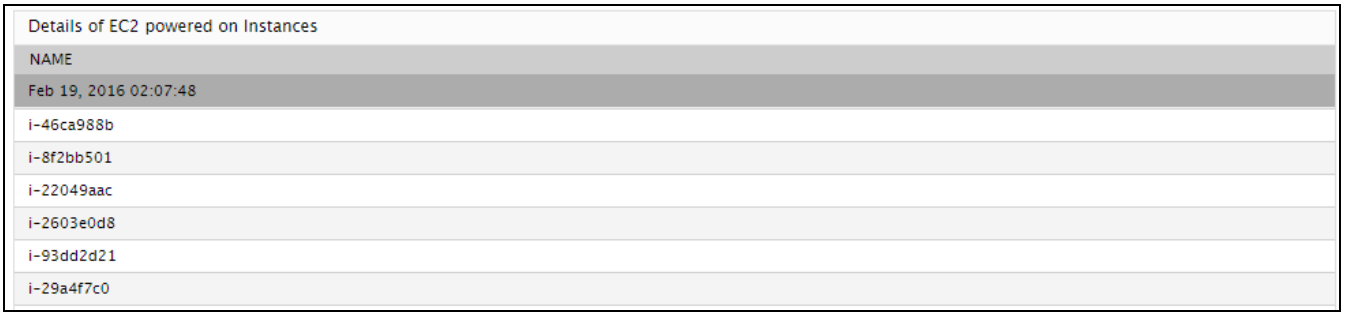


Figure 4.9: The detailed diagnosis of the EC2 instances poweredon measure

The detailed diagnosis of the **EBS volumes** measure displays the names of volumes that are available for use presently.

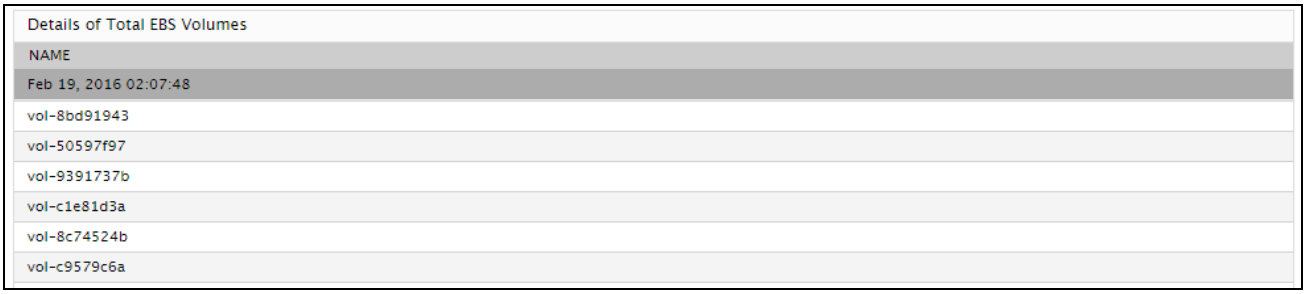


Figure 4.10: The detailed diagnosis of the EBS volumes measure

The detailed diagnosis of the **RDS instances** measure provides the details of all RDS instances configured for the AWS user account.

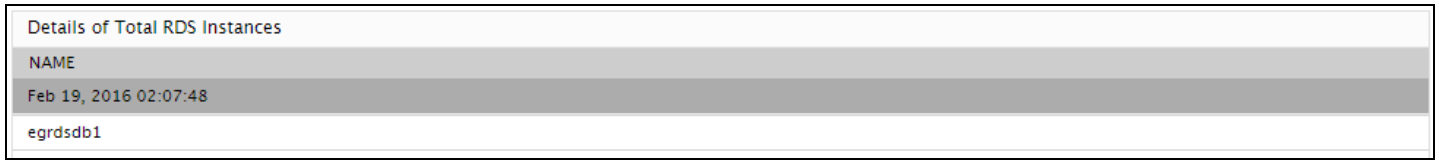


Figure 4.11: The detailed diagnosis of the RDS instances measure

The detailed diagnosis of the **RDS instances available** measure displays the details of the powered-on instances alone.

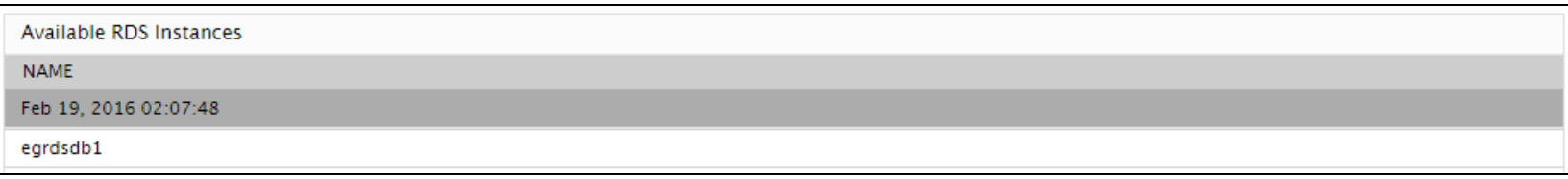


Figure 4.12: The detailed diagnosis of the RDS instances available measure

4

Administering the eG Manager to Monitor the AWS EC2 Region

To achieve this, follow the steps given below:

1. Log into the eG administrative interface.
2. eG Enterprise automatically discovers the AWS EC2 Regions once you set the ***Discover AWS EC2 cloud regions*** parameter to ***Yes*** while discovering the AWS EC2 Cloud. If the AWS EC2 Region is already discovered, use the Infrastructure -> Components ->Manage/Unmanage menu to manage it.
3. To manage the discovered components, go to the Infrastructure -> Components -> Manage/Unmanage page. The process of managing a component is clearly depicted by Figure 4.13 below.

   Note:

   For a more detailed procedure for managing components, refer to **Configuring and Monitoring Web Servers** document.

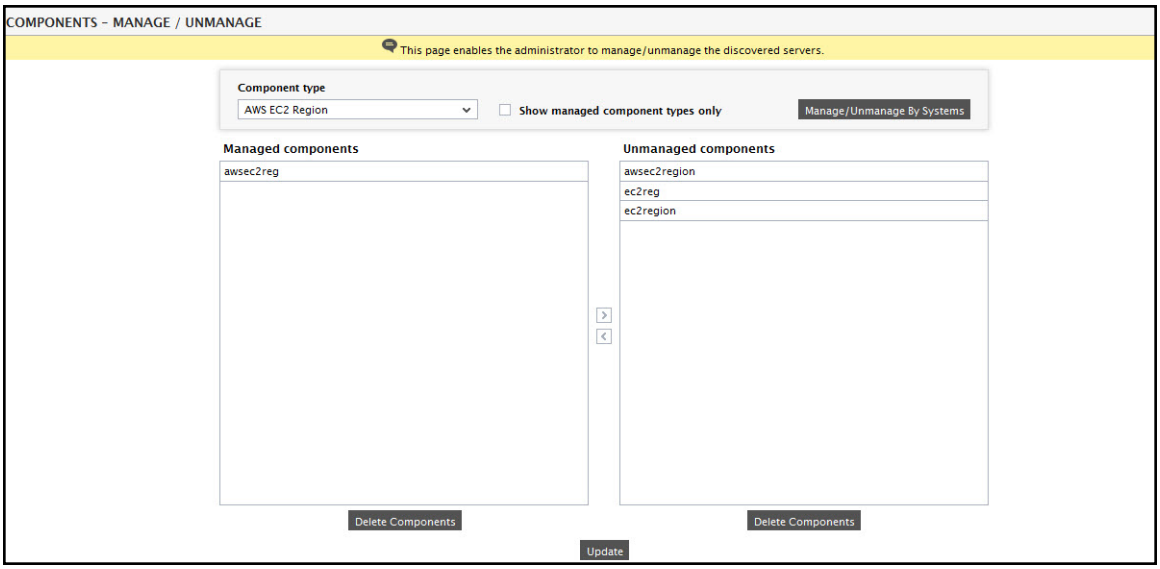


Figure 4.13: Managing an AWS EC2 Region

4. If you wish to manually add the Amazon EC2 Region component using the menu sequence: Infrastructure -> Components -> Add/Modify, ensure that you provide a valid name in the HOST text box instead of an IP address. In order to provide a valid name, ensure that the AllowQualifiedHostnames flag is set to **yes** in the **eg_services.ini** file of the **<EG_INSTALL_DIR>\manager\config** directory. Remember that components manually added are managed automatically.

   Note:

   The **HOST** name must be provided in the following format: **AWS EC2 Cloud/Region name**. In this case, the **HOST** name must be given as **awsec2cld/ap-southeast-1**

5. Now, when you attempt to sign out of the eG administrative interface, Figure 4.14 appears, listing the tests requiring manual configuration.

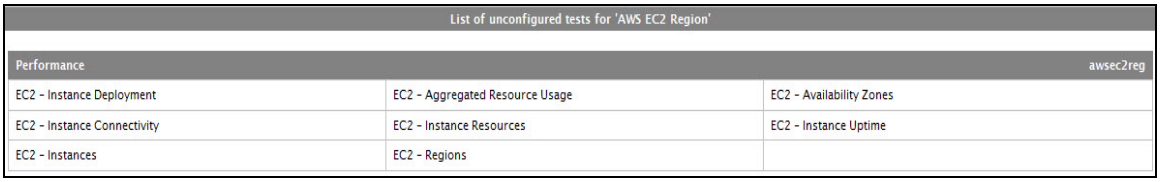


Figure 4.14: The list of unconfigured tests for AWS EC2 Region

6. Click on the **EC2 - Region** test to configure it. This test reports the availability of the Region and enables the administrator to figure out the time taken by the Region to respond to responses. To know how to configure the test, Click Here.

7. Finally, sign out of the eG administrative interface.

5

Monitoring the AWS EC2 Region

Amazon EC2 provides the ability to place instances in multiple locations. Amazon EC2 locations are composed of Availability Zones and Regions. Regions are dispersed and located in separate geographic areas (US, EU, etc.). Availability Zones are distinct locations within a Region that are engineered to be isolated from failures in other Availability Zones and provide inexpensive, low latency network connectivity to other Availability Zones in the same Region. By launching instances in separate Regions, you can design your application to be closer to specific customers or to meet legal or other requirements.

The *AWS EC2 Region* model offered by eG Enterprise monitors a specific region on the cloud and reports the availability and responsiveness of that region.

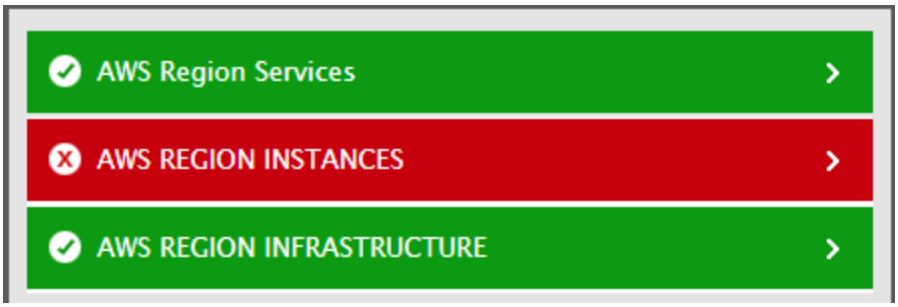


Figure 5.1: The layer model of the AWS EC2 Region

In addition, using a single eG agent installed on a remote Windows host in the environment, the model auto-discovers the IP address and the operating system of the instances launched on the cloud, periodically checks the powered-on status of each of the instances, continously assesses how each instance is utilizing the allocated resources, and thus promptly alerts you to unavailable and resource-hungry instances. As the solution also automatically determines what applications have been deployed on the instances, whenever one of these applications experience slowdowns, administrators can use the eG solution to instantly and accurately diagnose the root-cause of the slowdown - is it owing to the corresponding instance being unavailable or the application being resource-hungry?

Using the metrics so reported, administrators can ascertain the following:

- Is web-based (HTTP/HTTPS) access to the region available?
- Does it take an unreasonably long time to establish contact with the region?
- How many availability zones exist in the monitored region? What are they?
- Is any availability zone currently unavailable? If so, which one is it?
- Are all instances launched in the region accessible over the network?

Are any instances powered off currently?

- Were any instances launched/removed recently? If so, which ones are these?
- What type of instances are resource-intensive?

- Is any particular instance consuming too much CPU?
- Is the network traffic to/from any instance unusually high?
- Is the disk I/O of instances optimal?
- Was any instance rebooted recently? If so, which one is it?

Note:

The eG agent reports metrics for only availability zones and instances in a region that the configured AWS user account is allowed to access.

Some tests require the **AWS CloudWatch** service to be enabled. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. For enabling this service, you need to pay CloudWatch fees. Refer to the AWS web site for the fee details.

The sections that follow will discuss each layer of Figure 5.1 elaborately.

5.1 The AWS Region Infrastructure Layer

Using the tests mapped to this layer, you can promptly detect the non-availability of a target region and the availability zones in that region, and connection bottlenecks experienced while connecting to the cloud or its components.

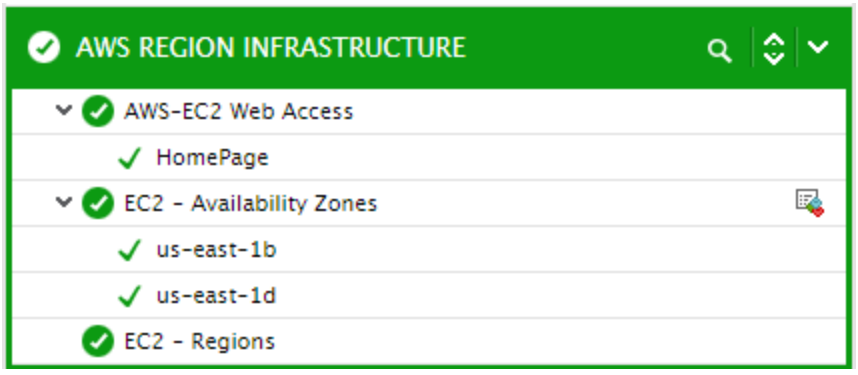


Figure 5.2: The tests mapped to the AWS Region Infrastructure layer

5.1.1 AWS EC2 - Web Access Test

This test emulates a user accessing a web page on the cloud via HTTP(S), and reports whether that page is accessible or not. In the process, the test indicates the availability of the cloud over the web, and the time it took for the agent to access the cloud over the web. This way, issues in web-based access to the cloud come to light.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for every URL configured for monitoring

Test parameters:

1. **TEST PERIOD** – How often should the test be executed
2. **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. By default, the url parameter is set to *HomePage:http://aws.amazon.com/ec2/* , where *HomePage* is the **URL name** , and *http://aws.amazon.com/ec2*is the **URL** value. You can modify this default setting to configure any **URL** of your choice - eg., the URL of the login page to your cloud-based infrastructure.
3. **HOST** - The host for which the test is to be configured.
4. **PORT** - The port to which the specified **HOST** listens
5. **COOKIEFILE** – Whether any cookies being returned by the web server need to be saved locally and returned with subsequent requests
6. **PROXYHOST** – The host on which a web proxy server is running (in case a proxy server is to be used)
7. **PROXYPORT** – The port number on which the web proxy server is listening
8. **PROXYUSERNAME** – The user name of the proxy server
9. **PROXYPASSWORD** – The password of the proxy server
10. **CONFIRM PASSWORD**– Confirm the password by retyping it here.
11. **CONTENT** – 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 test 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 test'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*.
12. **CREDENTIALS**– The HttpTest supports HTTP authentication. The **CREDENTIALS**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. These parameters will take either of the following values:
    - 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

      Where NTLM (Integrated Windows) authentication is supported, valid CREDENTIALS are mandatory. In other words, a *none* specification will not be supported in such cases. Therefore, in this case, against each configured URLname, 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 this test.

13. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
13. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none** , indicating that the proxy sever does not require authentication by default.
14. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
15. **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.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Availability:	Indicates whether the test was able to access the configured URL or not	Percent	Availability failures could be caused by several factors such as the web server process(es) (hosting the configured web page) 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 test. A response code between 200 to 300 indicates that the configured web page is available.
Total response time:	Indicates the time taken by the test to access this URL	Secs	Response time being high denotes a problem. Poor response times may be due to an

Measurement	Description	Measurement Unit	Interpretation
			overload. If the URL accessed involves the generation of dynamic content, 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:	Indicates whether the test managed to establish a TCP connection to this URL.	Percent	Failure to establish a TCP connection may imply that either the web server process hosting the web page 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 web page may start functioning properly again.
Tcp connect time:	Quantifies the time for establishing a TCP connection to the configured URL.	Secs	Typically, the TCP connection establishment must be very small (of the order of a few milliseconds).
Server response time:	Indicates the time period between when the connection was established and when the test 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).
Response code:	Returned by the test 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 test.	Kbytes	Typically the content length returned by the test for a specific URL should be the same across time. Any change in this metric may indicate the need for further investigation.
Content validity:	Validates whether the	Percent	A value of 100% indicates that the content

Measurement	Description	Measurement Unit	Interpretation
	test was successful in executing the request made to it.		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.

5.1.2 EC2 - Availability Zones Test

Amazon has data centers in different areas of the world (e.g., North America, Europe, Asia, etc.). Correspondingly, EC2 is available to use in different *Regions*. Each Region contains multiple distinct locations called *Availability Zones* (illustrated in the following diagram). Each Availability Zone is engineered to be isolated from failures in other Availability zones and to provide inexpensive, low-latency network connectivity to other zones in the same Region. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location.

If users complaint that their server instances are inaccessible, you may want to know whether it is because of the non-availability of the availability zone within which the instances have been launched. This test auto-discovers the availability zones configured within the monitored EC2 region, and reports the availability of each zone.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for each availability zone in the AWS EC2 Region being monitored

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured

3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED** - **This flag applies to the AWS-EC2 VM Resource Usage and aws-ec2 vm Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees**. For the fee details, refer to the AWS EC2 web site.
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

10. **EXCLUDE INSTANCE** - **This parameter applies only to EC2 - Instance Connectivity, EC2 - Instance Resources , EC2 - Instance Uptime, and EC2 - Instance Deployment tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.
11. **REPORT INSTANCE DATACENTER** - By default, this test reports the availability of only those availability zones that contain one/more instances. Accordingly, this flag is set to **true** by default. If you want the test to report metrics for all availability zones, regardless of whether/not they host instances, set this flag to **false**.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Availability:	Indicates whether/not this availability zone is currently available.	Number	The value 0 indicates that the availability zone is Not Available and the value 100 indicates that it is Available. If an availability zone fails, then all server instances operating within that zone will also be rendered unavailable. If you host all your Amazon EC2 instances in a single location that is affected by such a failure, your instances will be unavailable, thereby bringing your entire application to a halt. On the other hand, if you have instances distributed across many Availability Zones and one of the instances fails, you can design your application so the instances in the remaining Availability Zones handle any requests.

5.1.3 EC2 - Regions Test

Amazon EC2 provides the ability to place instances in multiple locations. Amazon EC2 locations are composed of Availability Zones and Regions. Regions are dispersed and located in separate geographic areas (US, EU, etc.). Each Region is completely independent.

By launching instances in separate Regions, you can design your application to be closer to specific customers or to meet legal or other requirements.

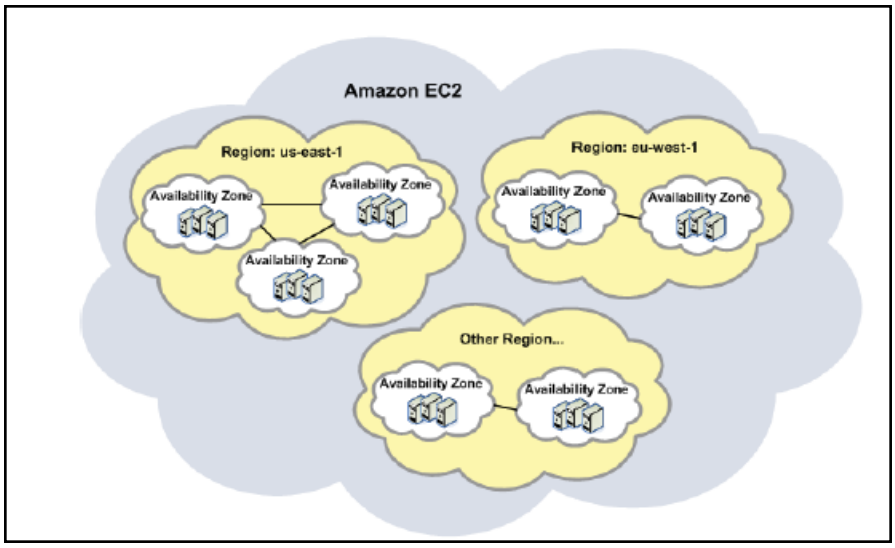


Figure 5.3: Regions and Availability zones

If a region is unavailable, then users to that region will not be able to access the server instances launched in that region. This may, in turn, adversely impact the user experience with the cloud. To avoid such an unpleasant outcome, it is best to periodically monitor the availability of each region, so that unavailable regions can be quickly and accurately identified, and the reasons for their non-availability remedied.

This test performs periodic availability checks on the monitored region, and reports the status of that region. In addition, the test also indicates the time taken for connecting to the region so that, connectivity issues can be isolated.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for the AWS EC2 region being monitored

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.

- From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
- Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED** - **This flag applies to the AWS-EC2 VM Resource Usage and aws-ec2 vm Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees. For the fee details, refer to the AWS EC2 web site**.
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **EXCLUDE INSTANCE** - **This parameter applies only to EC2 - Instance Connectivity, EC2 - Instance Resources , EC2 - Instance Uptime, and EC2 - Instance Deployment tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Availability:	Indicates whether/not the region is currently available.	Number	The value 0 indicates that the region is Not Available and the value 100 indicates that it is Available.
Response time:	Indicates the time taken to connect to the region.	Secs	A low value is typically desired for this measure. A high value or a consistent increase in this value could be indicative of connection bottlenecks.

5.2 The AWS Region Instances Layer

To determine issues in accessibility server instances launched in a region, and to detect the current state of each instance, use the tests mapped to this layer. The tests also auto-discover the server instances that are available (for the configured AWS user account) in a region, and report the uptime and the resource usage of the individual instances. Resource-hungry instances and those that were recently rebooted can thus be isolated.

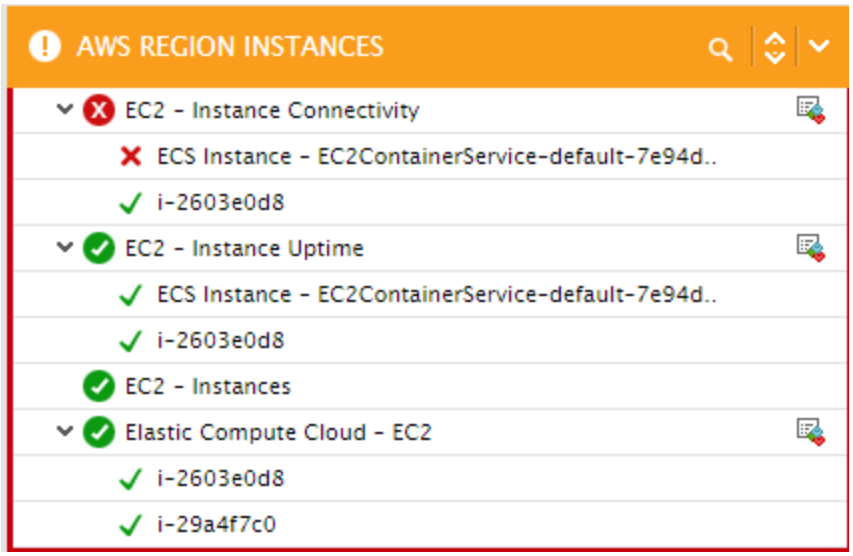


Figure 5.4: The tests mapped to the AWS EC2 Region Instance Status layer

5.2.1 Elastic Compute Cloud - EC2 Test

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. An EC2 instance is a virtual server in Amazon's Elastic Compute Cloud (EC2) for running

applications on the Amazon Web Services (AWS) infrastructure. Since users may run mission-critical applications on these EC2 instances, high uptime of the EC2 instances is imperative to the uninterrupted functioning of these applications and to ensure 100% user satisfaction with this cloud-based service. AWS administrators therefore, should frequently perform health checks on every instance, measure its load and resource usage, and capture potential failures and resource contentions, well before end-users notice and complain. This is exactly where the Elastic Compute Cloud - EC2 test helps!

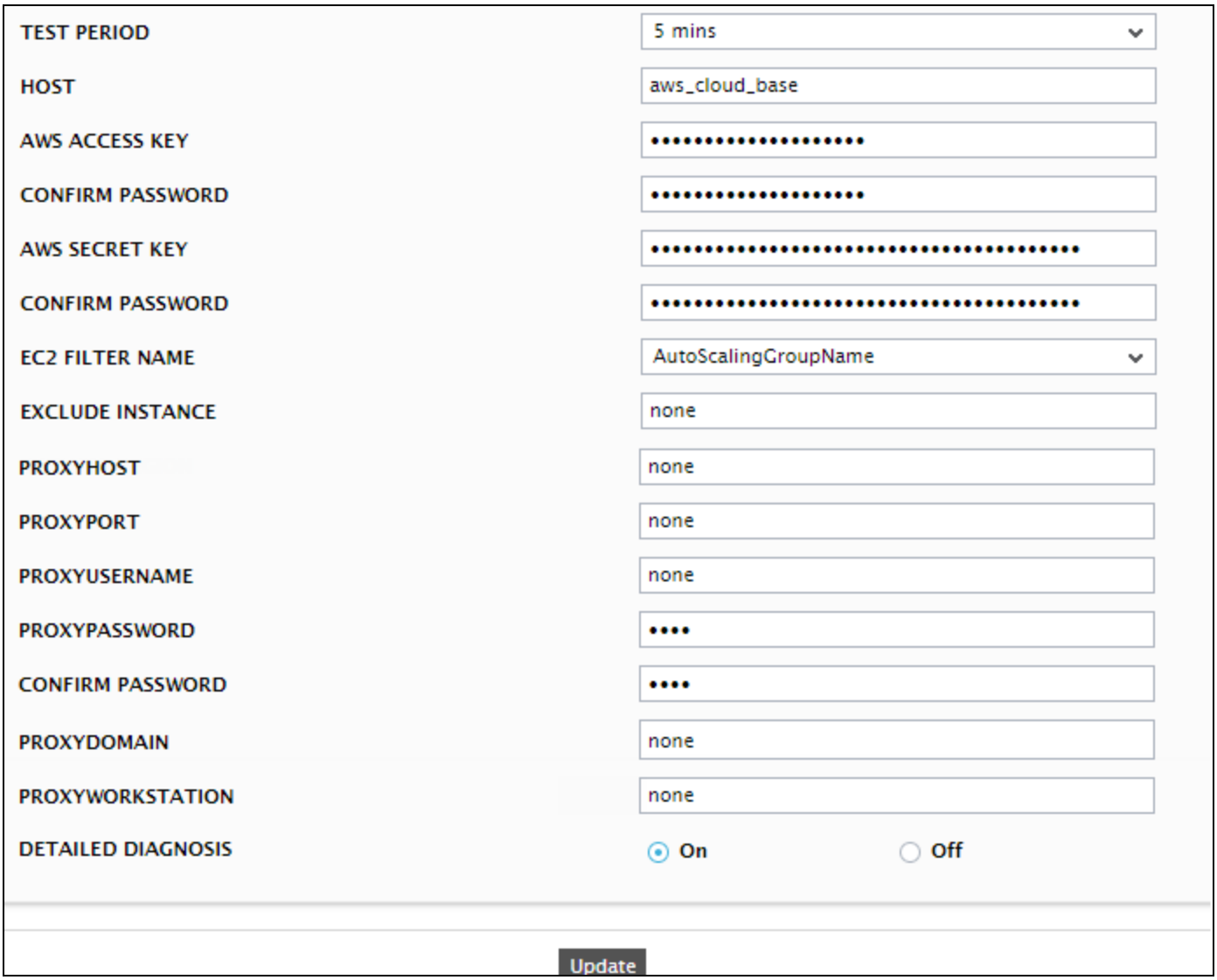
This test monitors the powered-on state of each EC2 instance and promptly alerts administrators if any instance has been powered-off inadvertently. Additionally, the test also reveals how each instance uses the CPU, disk, and network resources it is configured with, thus providing early pointers to irregularities in instance sizing, and prompting administrators to make necessary amends. This way, the test makes sure that critical applications are always accessible to end-users and perform at peak capacity.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for each instance / auto scaling group / instance type / image ID in the region being monitored, depending upon the option chosen from the **EC2 FILTER NAME** drop-down

Test parameters:



1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".

- Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
7. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
8. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
9. **EC2 FILTER NAME** - By default, this test reports metrics for each instance in the AWS infrastructure. This is why, the EC2 Filter Name flag is set to Instance ID by default. Alternatively, you can configure this test to aggregate metrics across a chosen collection of instances, and report one set of metrics per collection. For this, you just need to pick an instance collection from the EC2 Filter Name drop-down. The options available are as follows:

    - *AutoScalingGroupName*: Your EC2 instances can be organized into Auto Scaling Groups so that they can be treated as a logical unit for the purposes of scaling and management. When you create a group, you can specify its minimum, maximum, and, desired number of EC2 instances.

      If you select the *AutoScalingGroupName* option from the **EC2 FILTER NAME** drop-down, then this test will collect metrics for each instance, aggregate the metrics on the basis of the Auto Scaling Groups to which the instances belong, and report metrics for each group.

    - *InstanceType*: Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instance types comprise varying combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your applications.

      If the *InstanceType* option is chosen from the **EC2 FILTER NAME** drop-down, then this test will collect metrics for each instance, aggregate the metrics on the basis of the instance type, and report metrics for each type.

    - *ImageId*: Instances are created from Amazon Machine Images (AMI). The machine images are like templates that are configured with an operating system and other software, which determine the user's operating environment.

If the *ImageId* option is chosen from the **EC2 FILTER NAME** drop-down, then this test will collect metrics for each instance, aggregate the metrics on the basis of the AMI using which the instances were created, and report metrics for each image ID.

10. **EXCLUDE INSTANCE** - **This parameter is applicable only if InstanceId is chosen from the EC2 Filter Name drop-down.**

    In this case, against **EXCLUDE INSTANCE**, you can provide a comma-separated list of instance IDs you do not want the test to monitor.

11. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Instance power-on state:	Indicates the current powered-on state of this instance.		This measure is reported only if InstanceID is the option from the EC2 Filter Name drop-down of this test. The values that this measure can report and their corresponding numeric values are detailed in the table below:

Measure Value	Description	Numeric Value
Running	When the instance is ready for you, it enters the running state.	1
Pending	When you launch an instance, it enters the pending state	2

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td></td><td></td><td>
<table>
<tr><td>Terminated</td><td>When you no longer need an instance, you can terminate it, then it goes to terminated state.</td><td>3</td></tr>
<tr><td>Shutting down</td><td>While terminate the instance, As soon as the status of an instance changes to shutting- down or terminated</td><td>4</td></tr>
<tr><td>Stopping</td><td>When you stop your instance, it enters the stopping state</td><td>5</td></tr>
<tr><td>Stopped</td><td>After exiting the stopping state, it enters the stopped state</td><td>0</td></tr>
</table>
Note:
By default, this measure will report the Measure Values listed in the table above to indicate the current powered-on state of an instance. In the graph of this measure however, the same will be represented using the numeric equivalents only.
</td></tr>
<tr><td>EBS volumes</td><td>Indicates the number of EBS volumes attached to this instance.</td><td>Number</td><td>This measure is reported only if the InstanceId option is chosen from the EC2 Filter Name drop-down of this test.
You can attach an EBS volumes to one of your instances that is in the same Availability Zone</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
			as the volume. You can attach multiple volumes to the same instance within the limits specified by your AWS account. Your account has a limit on the number of EBS volumes that you can use, and the total storage available to you. Using the detailed diagnosis of this measure, you can identify the volumes that are attached to this EC2 instance.
CPU credit usage:	Indicates the number of CPU credits consumed by this T2 instance / all T2 instances / all T2 instances created from this image ID during the last measurement period.	Number	This measure is reported only for individual T2 instances, the T2 instance type, and the image ID using which T2 instances (if any) were created. A CPU Credit provides the performance of a full CPU core for one minute. Traditional Amazon EC2 instance types provide fixed performance, while T2 instances provide a baseline level of CPU performance with the ability to burst above that baseline level. The baseline performance and ability to burst are governed by CPU credits. One CPU credit is equal to one vCPU running at 100% utilization for one minute. Other combinations of vCPUs, utilization, and time are also equal to one CPU credit; for example, one vCPU running at 50% utilization for two minutes or two vCPUs running at 25% utilization for two minutes. Each T2 instance starts with a healthy initial CPU credit balance and then continuously (at a millisecond-level resolution) receives a set rate of CPU credits per hour, depending on instance size. When a T2 instance uses fewer CPU resources than its base performance level

Measurement	Description	Measurement Unit	Interpretation
			allows (such as when it is idle), the unused CPU credits (or the difference between what was earned and what was spent) are stored in the credit balance for up to 24 hours, building CPU credits for bursting. When your T2 instance requires more CPU resources than its base performance level allows, it uses credits from the CPU credit balance to burst up to 100% utilization. The more credits your T2 instance has for CPU resources, the more time it can burst beyond its base performance level when more performance is needed. This implies that ideally, the value of the CPU credit usage measure should be low for an instance and the value of the CPU credit balance for that instance should be high, as that way, an instance is assured of more CPU resources when performance demands increase. By comparing the value of this measure across instances, you can precisely identify the instance that has used up a sizeable portion of its CPU credits.
CPU credit balance:	Indicates the number of CPU credits that have been earned by this T2 instance / all T2 instances / all T2 instances created from this image ID	Number	
Disk read operations:	Indicates the rate at which read operations were performed on all disks available to this instance.	Operations/Sec	Compare the value of this measure across instances to know which instance is too slow in processing read requests.
Disk write operations:	Indicates the rate at which write operations were performed on all disks available to this instance.	Operations/Sec	Compare the value of this measure across instances to know which instance is too slow in processing write requests.
Disk reads:	Indicates the rate at which data was read from all disks	KB/Sec	Compare the value of this measure to identify the instance that is the slowest in responding to read requests.

Measurement	Description	Measurement Unit	Interpretation
	available to this instance.		
Disk writes:	Indicates the rate at which data was written to all disks available to this instance.	KB/Sec	Compare the value of this measure to identify the instance that is the slowest in responding to write requests.
Incoming network traffic:	Indicates the rate at which data was received by all network interfaces of this instance.	KB/Sec	Compare the value of these measures across instances to know which instance is consuming too much bandwidth. Then, compare the value of the Incoming network traffic and Outgoing network traffic measures of that instance to determine where bandwidth consumption was more - when receiving data over the network? or when sending data?
Outgoing network traffic:	Indicates the rate at which data was sent by all the network interfaces of this instance.	KB/Sec	
EC2 status check:	Indicates whether a status check (system status check or instance status check) failed for this instance		Amazon EC2 performs automated checks on every running EC2 instance to identify hardware and software issues. These status checks are of two types: system and instance status checks. If either of these status checks fails, then this measure will report the value *Failed*. If none of these status checks fail, then this measure will report the value *Passed*. The values that this measure can report and their corresponding numeric values are listed in the table below: **Measure Value** **Numeric Value** Failed 1 Passed 0 **Note:**

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td></td><td></td><td>By default, this measure reports the Measure Values above to indicate whether a check passed or failed. In the graph of this measure however, the same is indicated using the numeric equivalents only.</td></tr>
<tr><td>EC2 instance status check:</td><td>Indicates whether/not this instance passed the EC2 instance status check in the last minute.</td><td></td><td>Instance status checks monitor the software and network configuration of your individual instance. These checks detect problems that require your involvement to repair. When an instance status check fails, typically you will need to address the problem yourself (for example, by rebooting the instance or by making instance configuration changes).

The following are examples of problems that can cause instance status checks to fail:

• Failed system status checks
• Incorrect networking or startup configuration
• Exhausted memory
• Corrupted file system
• Incompatible kernel

The values that this measure can report and their corresponding numeric values are listed in the table below:

<table><tr><th>Measure Value</th><th>Numeric Value</th></tr><tr><td>Failed</td><td>1</td></tr><tr><td>Passed</td><td>0</td></tr></table>
Note:

By default, this measure reports the Measure Values above to indicate whether a check passed or failed. In the graph of this measure however, the same is indicated using the numeric equivalents only.</td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
EC2 system status check:	Indicates whether/not this instance passed the EC2 system status check in the last minute.	Number	System status checks monitor the AWS systems required to use your instance to ensure they are working properly. These checks detect problems with your instance that require AWS involvement to repair. When a system status check fails, you can choose to wait for AWS to fix the issue, or you can resolve it yourself (for example, by stopping and starting an instance, or by terminating and replacing an instance). The following are examples of problems that can cause system status checks to fail: • **Loss of network connectivity** • **Loss of system power** • **Software issues on the physical host** • **Hardware issues on the physical host** The values that this measure can report and their corresponding numeric values are listed in the table below: **Measure Value** / **Numeric Value** Failed / 1 Passed / 0 **Note:** By default, this measure reports the **Measure Value**s above to indicate whether a check passed or failed. In the graph of this measure however, the same is indicated using the numeric equivalents only.

Detailed Diagnosis:

Using the detailed diagnosis of the **EBS volumes** measure, you can identify the volumes that are attached to a particular EC2 instance.

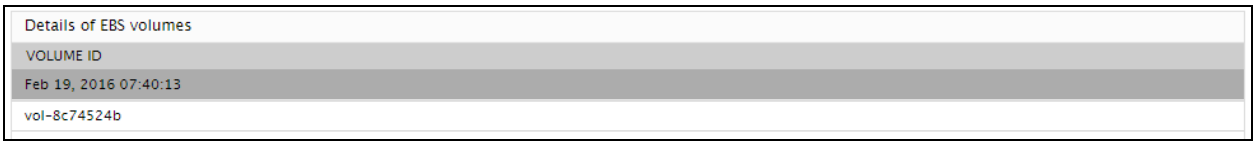


Figure 5.5: The detailed diagnosis of the EBS volumes measure

5.2.2 EC2 - Instance Uptime Test

In cloud-based environments, it is essential to monitor the uptime of server instances launched on the cloud. By tracking the uptime of each of the instances, administrators can determine what percentage of time an instance has been up. Comparing this value with service level targets, administrators can determine the most trouble-prone areas of the infrastructure hosted on the cloud.

In some environments, administrators may schedule periodic reboots of their instances. By knowing that a specific instance has been up for an unusually long time, an administrator may come to know that the scheduled reboot task is not working on an instance.

This test monitors the uptime of each instance available to the configured AWS user account.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for each instance launched by the configured AWS user account in the monitored region

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with

the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and AWS-EC2 VM Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees**. **For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **EXCLUDE INSTANCE** - **This parameter applies only to EC2 - Instance Connectivity, EC2 - Instance Resources , EC2 - Instance Uptime, and EC2 - Instance Deployment tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.
11. **REPORT MANAGER TIME** - By default, this flag is set to **Yes**, indicating that, by default, the detailed diagnosis of this test, if enabled, will report the shutdown and reboot times of the cloud in the manager's time zone. If this flag is set to **No**, then the shutdown and reboot times are shown in the time zone of the system where the agent is running (i.e., the system system on which the remote agent is running).
12. **DD FREQUENCY** - Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is 1:1. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this

frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against DD frequency.

13. **DETAILED DIAGNOSIS**- To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Has the instance been rebooted?:	Indicates whether this instance has been rebooted during the last measurement period or not.	Boolean	If this measure shows 1, it means that the instance was rebooted during the last measurement period. By checking the time periods when this metric changes from 0 to 1, an administrator can determine the times when this instance was rebooted.
Uptime of the instance during the last measure period:	Indicates the time period that the instance has been up since the last time this test ran.	Secs	If the instance has not been rebooted during the last measurement period and the agent has been running continuously, this value will be equal to the measurement period. If the instance 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 instance 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.
Total uptime of the instance:	Indicates the total time that this intance has been up since its last reboot.	Mins	Administrators may wish to be alerted if an instance has been running without a reboot for a very long period. Setting a threshold for this metric allows administrators to determine such conditions.

Detailed Diagnosis:

The detailed diagnosis of the *Has VM been rebooted?* measure reveals when the instance was last shutdown, when it was rebooted, how long the shutdown lasted, and whether the instance was shutdown as part of a routine maintenance exercise.

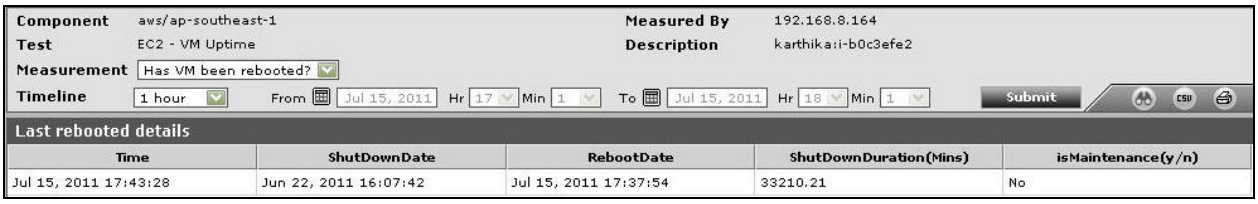


Figure 5.6: The detailed diagnosis of the Has VM been rebooted? measure

5.2.3 EC2 - Instances Test

An Amazon Machine Image (AMI) contains all information necessary to boot instances of your software. For example, an AMI might contain all the software to act as a web server (e.g., Linux, Apache, and your web site) or it might contain all the software to act as a Hadoop node (e.g., Linux, Hadoop, and a custom application). After an AMI is launched, the resulting running system is called an **instance**. All instances based on the same AMI start out identical and any information on them is lost when the instances are terminated or fail.

Users with valid AWS user accounts can sign into an EC2 region to view and use available instances, or purchase and launch new ones. With the help of this test, you can determine the total number of instances that are currently available for the configured AWS user account in the monitored region, the number of instances that were newly purchased/terminated, and the count of powered-off instances.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for the AWS EC2 Region being monitored

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.

- Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
- Once the payment is made, the user will be automatically signed in to the AWS account.
- From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
- Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and aws-ec2 vm Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees. For the fee details, refer to the AWS EC2 web site**.
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **EXCLUDE INSTANCE** - **This parameter applies only to EC2 - Instance Connectivity, EC2 - Instance Resources , EC2 - Instance Uptime, and EC2 - Instance Deployment tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.

11. **DETAILED DIAGNOSIS**- To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Total instances:	Indicates the total number of instances currently available for the configured AWS user account.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the instances available for use for the configured AWS account, regardless of the current state of the instances.
Instances powered on:	Indicates the total number of instances that are currently powered-on.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the powered-on instances available for use for the configured AWS account.
Instances powered off:	Indicates the total number of instances that are currently powered-off.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the powered-off instances available for the configured AWS account.
Added instances:	Indicates the total number of instances that were newly purchased by the configured AWS user account during the last measurement period.	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the instances that were newly purchased and launched by the configured AWS user account.
Removed instances:	Indicates the total number of instances that were newly terminated by the	Number	The detailed diagnosis capability of this measure, if enabled, shows the details of all the instances that were newly terminated/removed by the configured AWS

Measurement	Description	Measurement Unit	Interpretation
	configured AWS user account during the last measurement period.		user account.

Detailed Diagnosis:

The detailed diagnosis capability of the *Total instances* measure, if enabled, shows the details of all the instances available for use for the configured AWS account in the monitored region, regardless of the current state of the instances.

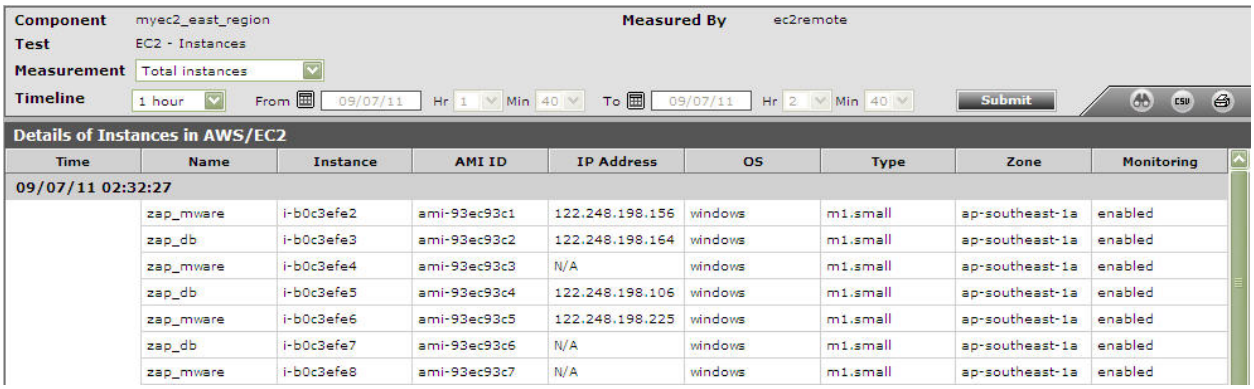


Figure 5.7: The detailed diagnosis of the Total instances measure

The detailed diagnosis capability of the *Instances powered on* measure, if enabled, shows the details of all the powered-on instances available for use for the configured AWS account in the monitored region.

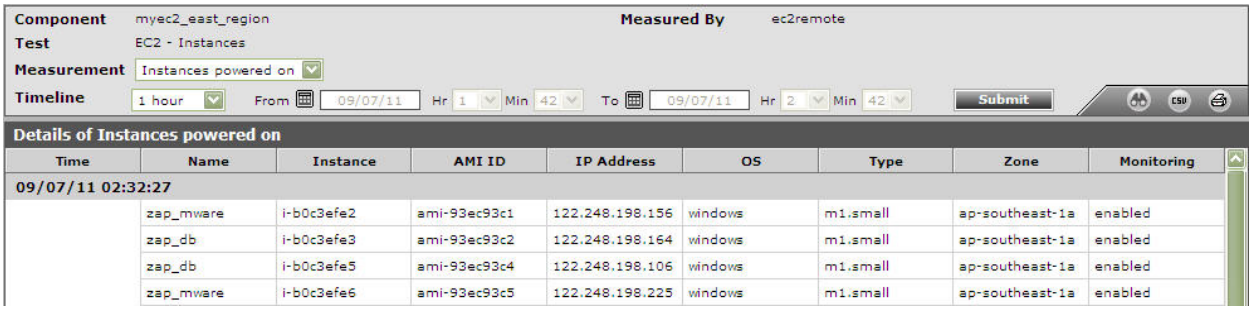


Figure 5.8: The detailed diagnosis of the Instances powered on measure

The detailed diagnosis capability of the *Instances powered off* measure, if enabled, shows the details of all the powered-off instances available for the configured AWS account.

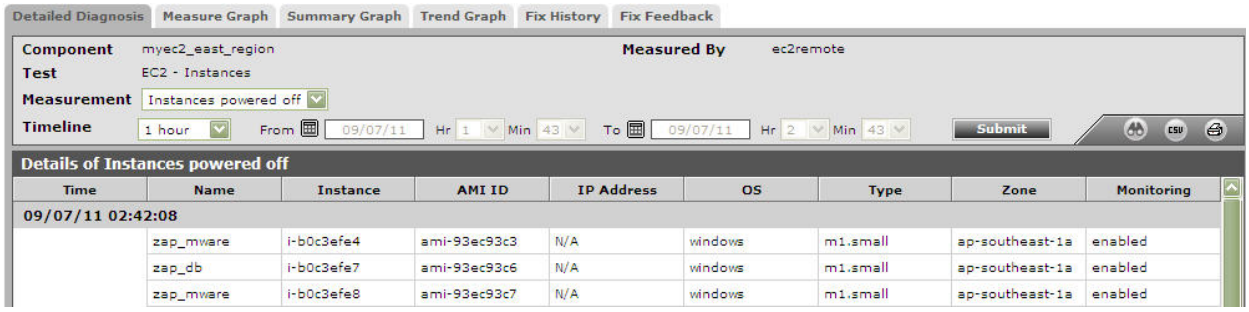


Figure 5.9: The detailed diagnosis of the Instances powered off measure

5.2.4 EC2 - Instance Resources Test

Tracking the CPU usage, disk and network I/O of every instance launched by a configured AWS user account in a region will provide administrators with valuable insights into how well the instances are utilizing the allocated resources. The **EC2 - Instance resources** test does just that. This test auto-discovers the instances available for the configured AWS user account in a region, and reports the resource usage of each instance so that, administrators can quickly compare the usage metrics across instances and pinpoint which instance is resource-hungry.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for each instance launched by the configured AWS user account in the monitored region

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with

the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **CLOUDWATCH ENABLED**- **This flag applies to the AWS-EC2 VM Resource Usage and AWS-EC2 VM Aggregate Resource usage tests only**. These tests report critical metrics pertaining to the resource usage of the server instances launched in the cloud. If you want these tests to report resource usage metrics very frequently - say, once every minute or lesser - you will have to configure the tests to use the **AWS CloudWatch** service. This is a **paid** web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics. To enable the above-mentioned tests to use this service, set the cloudwatch enabled flag to *true*. On the other hand, to report resource usage metrics less frequently - say, once in 5 minutes or more - these tests do not require the **AWS CloudWatch** service; in this case therefore, set the cloudwatch enabled flag to *false*. **Note that for enabling CloudWatch, you will have to pay CloudWatch fees**. **For the fee details, refer to the AWS EC2 web site.**
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
10. **EXCLUDE INSTANCE** - **This parameter applies only to EC2 - Instance Connectivity, EC2 - Instance Resources , EC2 - Instance Uptime, and EC2 - Instance Deployment tests.** In the **EXCLUDE INSTANCE** text box, provide a comma-separated list of instance names or instance name patterns that you do not wish to monitor. For example: i-b0c3e*,*7dbe56d. By default, this parameter is set to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
CPU utilization:	Indicates the percentage of allocated CPU consumed by this instance.	Percent	A high value for this measure indicates that an instance is utilizing CPU excessively - this could be because of one/more resource-intensive processes executing on that instance. Compare the value of this measure across instances to identify the CPU- intensive instances.
Incoming network traffic:	Indicates the rate of incoming network traffic i.e., the rate at which the bytes are received by all the network interfaces connected to this instance.	KB/Sec	Compare the values of these measures across instances to quickly identify the instance that is utilizing the network bandwidth excessively.
Outgoing network traffic:	Indicates the volume of outgoing network traffic i.e., the rate at which the bytes are transferred from all the network interfaces connected to this instance.	KB/Sec	
Disk reads:	Indicates the rate at which data is read from the disks of this instance.	KB/Sec	These measures are good indicators of the level of disk I/O activity on an instance. By comparing the values of these measures across instances, you can accurately determine which instance is performing I/O-intensive operations.
Disk writes:	Indicates the rate at which data is written to the disks of this instance.	KB/Sec	
Disk read operations:	Indicates the rate at	Operations/Sec	These measures are good indicators of the

Measurement	Description	Measurement Unit	Interpretation
	which disk read operations are performed on this instance.		level of disk I/O activity on an instance type. By comparing the values of these measures across types, you can accurately determine the type of instances that is performing I/O-intensive operations.
Disk write operations:	Indicates the rate at which disk write operations were performed on this instance.	Operations/Sec	

5.3 The AWS EC2 Region Services Layer

The tests mapped to this layer auto-discover the server instances that are available (for the configured AWS user account) in a region, and reports the uptime and the resource usage of the individual instances. Resource-hungry instances and those that were recently rebooted can thus be isolated.

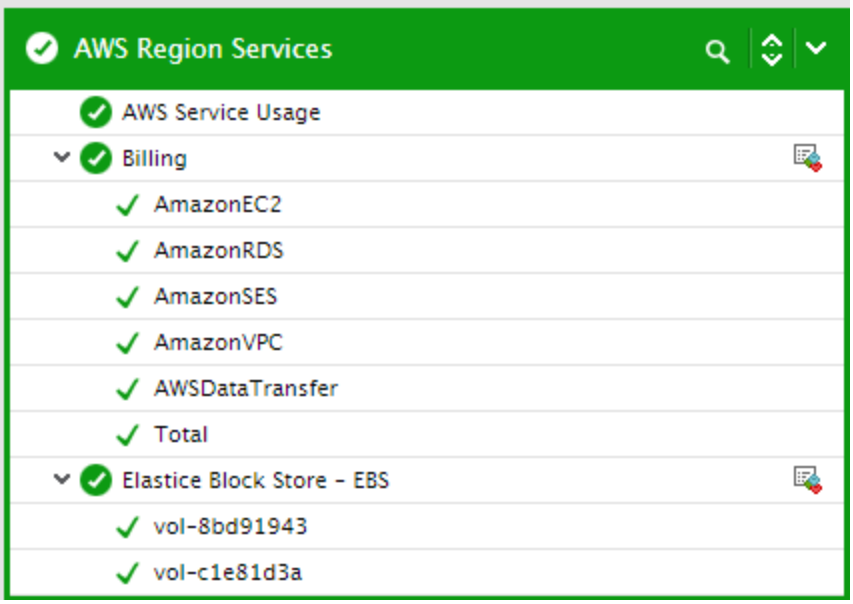


Figure 5.10: The tests mapped to the AWS EC2 Region Instance Details layer

5.3.1 AWS Service Usage Test

Use this test to receive an overview of the instances launched and services (EBS and RDS) used by a configured AWS user account in a monitored region. Understanding how many instances of a service are utilized by an account will help you to bill that user accordingly.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for the AWS account configured for monitoring

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
7. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.

8. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
9. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
EC2 instances:	Indicates the total number of EC2 instances currently available for the configured AWS user account in the monitored region.	Number	Use the detailed diagnosis of this measure to know which are the available instances.
EC2 instances poweredon:	Indicates the total number of instances that are currently powered- on for the configured AWS user account in the monitored region.	Number	Use the detailed diagnosis of this measure to know the names of the powered-on instances.
EBS volumes:	Indicates the total number of EBS volumes currently available for the configured AWS user account in the monitored region.	Number	Use the detailed diagnosis of this measure to know which volumes are available for use presently.

Measurement	Description	Measurement Unit	Interpretation
RDS instances:	Indicates the total number of RDS instances that are configured for the AWS user account in the monitored region.	Number	Use the detailed diagnosis of this measure to know the details of all RDS instances configured for the AWS user account.
RDS instances available:	Indicates the number of instances that are currently powered-on and available for the use of the configured AWS user account in the monitored region.	Number	Use the detailed diagnosis of this measure to know the details of all RDS instances that are powered-on and available for the use of the AWS user account.

The detailed diagnosis of the **EC2 instances** measure lists the names of all EC2 instances available for the configured AWS user account.

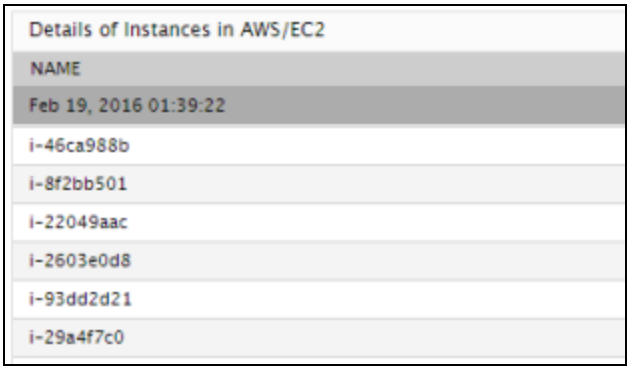


Figure 5.11: The detailed diagnosis of the EC2 instances measure

The detailed diagnosis of the **EC2 instances poweredon** measure lists the names of the powered-on instances alone.

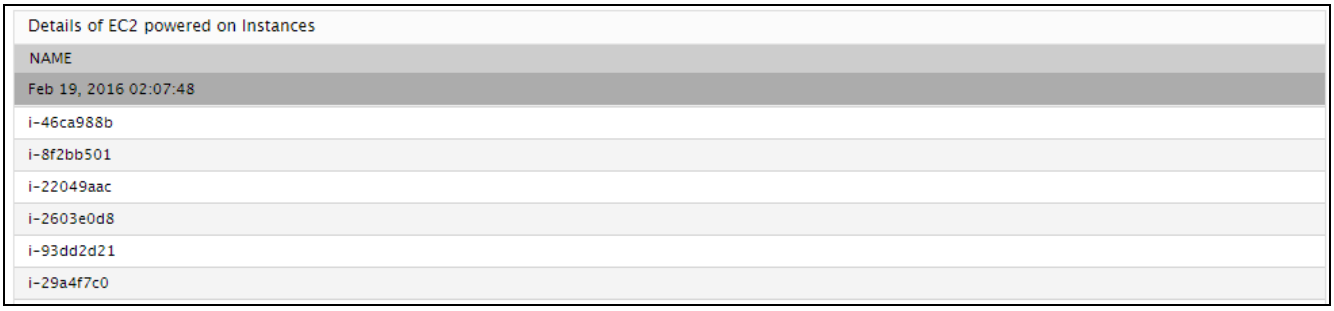


Figure 5.12: The detailed diagnosis of the EC2 instances poweredon measure

The detailed diagnosis of the **EBS volumes** measure displays the names of volumes that are available for use presently.

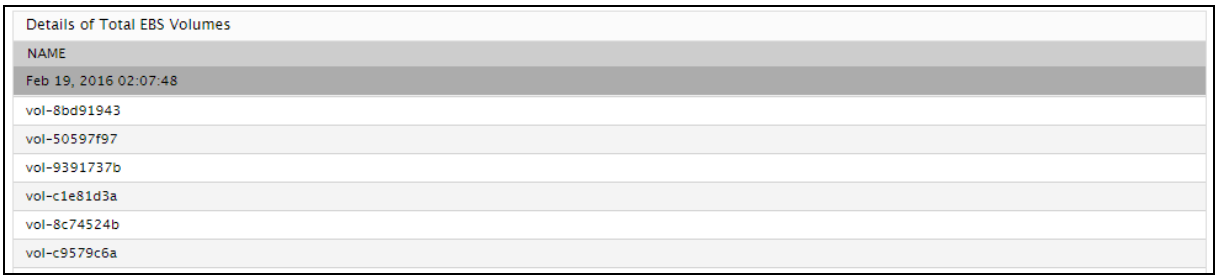


Figure 5.13: The detailed diagnosis of the EBS volumes measure

The detailed diagnosis of the **RDS instances** measure provides the details of all RDS instances configured for the AWS user account.

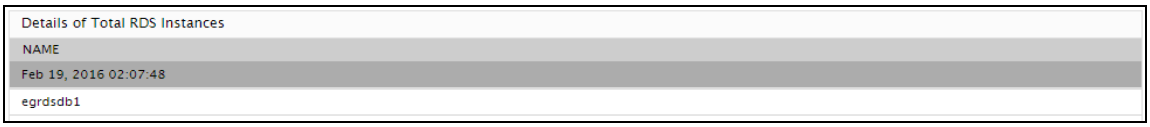


Figure 5.14: The detailed diagnosis of the RDS instances measure

The detailed diagnosis of the **RDS instances available** measure displays the details of the powered-on instances alone.

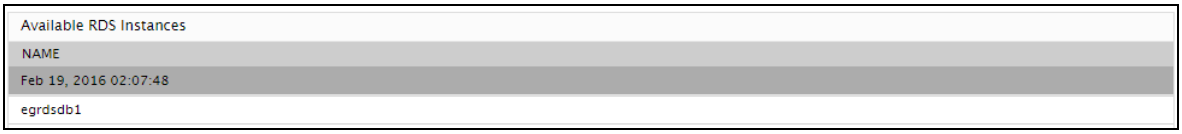


Figure 5.15: The detailed diagnosis of the RDS instances available measure

5.3.2 EC2 Container - ECS Test

AWS users can opt to run instances within Elastic Compute Cloud (EC2) or look into using containers. Amazon EC2 Container Service (ECS) manages Docker containers within AWS, allowing users to easily scale up or down and evaluate and monitor CPU usage. These AWS containers run on a managed cluster of EC2 instances, with ECS automating installation and operation of the cluster infrastructure. The first step to get started with ECS therefore is to create a cluster and launch EC2 instances in it. Then, create task definitions. A task is one or more Docker containers running together for one service or a microservice. When configuring a container in your task definition, you need to define the container name and also indicate how much memory and how many CPU units you want to reserve for each container. Finally, you will have to create a service, so that you can run and maintain a specified number of instances of a task definition simultaneously.

Time and again, administrators will have to check on the resource usage of each cluster, so that they can identify those clusters that have been consistently over-utilizing the CPU and memory resources. Resource usage at the individual service-level should also be monitored, so that administrators can figure out whether the excessive resource consumption by a cluster is because the cluster itself does not have enough resources at its disposal, or because one/more services running on the cluster are depleting the resources. Using the AWS EC2 Container - ECS test, administrators can monitor resource usage both at the cluster and the service-level.

This test auto-discovers the clusters configured in the region being monitored and also the services running on each cluster. CPU and memory usage is then reported for each cluster and service, alongside the CPU and memory reservations (of all tasks) per cluster. These insights help administrators understand where there is a contention for resources - at the cluster-level? or at the service-level? or both? - and accordingly decide what needs to be done to optimize resource usage:

- Should more container instances be added to the cluster to increase the amount of resources at its disposal?
- Should the task definitions of the resource-hungry services be fine-tuned so that the service has more resources to use?

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for each cluster:service pair in the monitored region

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.

6. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
7. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none** , indicating that the proxy sever does not require authentication by default.
8. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
9. **ECS FILTER** - By default, this test reports metrics for each cluster and for each service that is running on a cluster. Accordingly, *ServiceName* is the default selection from the **ECS FILTER** drop-down. If you do not want service-level metrics, then you can configure the test to report resource usage at the cluster-level alone. For this, just select *ClusterName* from the **ECS FILTER** drop-down. If this is done, then the test will only report cluster names as descriptors.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
CPU reservation:	The percentage of CPU units that are reserved by running tasks in this cluster.	Percent	**This measure is reported at the cluster- level only - i.e., for the ClusterName descriptor alone.** This value is computed using the following formula: *Total CPU units reserved by ECS tasks on the cluster / Total CPU units that were registered for all the container instances in the cluster * 100* A value close to 100% indicates that almost all resources available to the cluster are being reserved by running tasks in that cluster. This implies that additional services cannot be configured on that cluster until more resources are made available to the cluster or until the CPU reservation of running tasks is

Measurement	Description	Measurement Unit	Interpretation
			reduced.
CPU utilization:	Indicates the percentage of CPU units used by this cluster or by this service	Percent	For a cluster, this value is computed using the following formula: *Total CPU units currently used by ECS tasks on this cluster / Total CPU units that were registered for all the container instances in this cluster * 100* A value close to 100% for this measure at the cluster-level could either indicate that the cluster is resource-starved or that one/more services running on the cluster are consuming excessive resources. If the reason for high CPU usage is the poor resource configuration of the cluster, then, you may want to add more instances to the cluster to add to its resource base. On the other hand, if the cluster is adequately sized with CPU, then you may want to check the value of this measure for each of the services running on the cluster . For a service, this value is computed using the following formula: *Total CPU units currently used by ECS tasks defined for this service / Total CPU units that are reserved for the tasks defined for this service * 100* Compare the value of this measure across services of a cluster to know which services of that cluster are guilty of over-utilization of CPU. Once the services are identified, check the CPU reservation of the task definitions of those services to determine whether sufficient resources have been allocated to those tasks. If not, increase the reservations to allow optimal

Measurement	Description	Measurement Unit	Interpretation
			resource usage.
Memory reservation:	The percentage of memory that is reserved by running tasks in this cluster.	Percent	**This measure is reported at the cluster- level only - i.e., for the ClusterName descriptor alone.** This value is computed using the following formula: *Total amount of memory reserved by ECS tasks on the cluster / Total amount of memory that was registered for all the container instances in the cluster * 100* A value close to 100% indicates that almost all resources available to the cluster are being reserved by running tasks in that cluster. This implies that additional services cannot be configured on that cluster until more resources are made available to the cluster or until the memory reservation of running tasks is reduced.
Memory utilization:	Indicates the percentage of memory used by this cluster or by this service	Percent	For a cluster, this value is computed using the following formula: *Total memory currently used by ECS tasks on this cluster / Total memory that is registered for all the container instances in this cluster * 100* A value close to 100% for this measure at the cluster-level could either indicate that the cluster is resource- starved or that one/more services running on the cluster are consuming excessive resources. If the reason for high memory usage is the poor resource configuration of the cluster, then, you may want to add more instances to the cluster to add to its resource base. On the other hand, if the cluster is adequately sized with memory, then you

Measurement	Description	Measurement Unit	Interpretation
			may want to check the value of this measure for each of the services running on the cluster . For a service, this value is computed using the following formula: *Total memory currently used by ECS tasks defined for this service / Total memory reserved for the tasks defined for this service * 100* Compare the value of this measure across services of a cluster to know which services of that cluster are guilty of over-utilization of memory. Once the services are identified, check the memory reservation of the task definitions of those services to determine whether sufficient resources have been allocated to those tasks. If not, increase the reservations to allow optimal resource usage.

5.3.3 RedShift Test

Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud. The first step to create such a data warehouse is to launch an Amazon Redshift cluster. An Amazon Redshift cluster is a collection of computing resources called nodes. Each cluster runs an Amazon Redshift engine and contains one or more databases. Each cluster has a leader node and one or more compute nodes. The leader node receives queries from client applications, parses the queries, and develops query execution plans. The leader node then coordinates the parallel execution of these plans with the compute nodes, aggregates the intermediate results from these nodes, and finally returns the results back to the client applications. Compute nodes execute the query execution plans and transmit data among themselves to serve these queries. The intermediate results are sent back to the leader node for aggregation before being sent back to the client applications.

Where RedShift is in use, query performance, and consequently, the performance of the dependent client applications, depends upon the following factors:

- **Cluster availability**
- **How the cluster and its nodes use the CPU, network, and storage resources of the cluster;**
- **Responsiveness of the nodes in the cluster to I/O requests from client applications**

To be able to accurately assess whether cluster performance is at the desired level or not, an administrator would require real-time insights into each of the factors listed above. The RedShift test provides administrators with these valuable insights. By reporting the current health status of each cluster managed by RedShift, this test brings unavailable clusters to light. The resource usage of the cluster is also reported, so that potential resource contentions can be proactively isolated. Optionally, you can also configure this test to report metrics for individual nodes in the cluster as well. If this is done, then administrators will be able to instantly drill-down from a resource-hungry cluster to the exact node in the cluster that could hogging the resources. At the node-level, the latency and throughput of each node is also revealed. This way, when users complain of degradation in the performance of client applications, you can quickly identify the cluster and the precise node in the cluster that is slowing down I/O processing and consequently, impacting application performance.

Target of the test: Amazon EC2 Cloud

Agent deploying the test: A remote agentx

Output of the test:

One set of results for each cluster and/or node in the monitored AWS region

First level descriptor: Cluster

Second level descriptor: Node

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **REDSHIFT FILTER NAME** - By default, this test reports metrics only for each cluster in each AWS region on the cloud. This is why, this flag is set to **ClusterIdentifier**, by default. If needed, you can configure the test to additionally report metrics for every node in every cluster. For node-level metrics, select the **NodeIdentifer** option from this drop-down. Upon selection, you will be able to view metrics both at the cluster-level and the node-level.
7. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
8. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
9. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
CPU utilization:	Indicates the percentage of CPU utilized by this cluster/node.	Percent	For a cluster, this measure will report the aggregate CPU usage of all nodes in the cluster. If the value of this measure is consistently above 50% for a cluster, it indicates that a serious resource contention may occur on that cluster, if additional processing power is not provided to it. In such a case, you may want to consider adding more nodes to the cluster, or adding more CPUs to the existing nodes. You can also compare the CPU usage of nodes in the resource- hungry cluster to determine whether one/more nodes are hogging the CPU. If so, you may want to

Measurement	Description	Measurement Unit	Interpretation
			tweak the load-balancing algorithm of your cluster to ensure uniform load distribution.
Database connections:	Indicates the number of connections to the databases in this cluster.	Number	**This measure is only reported at the cluster-level and not the node-level.**
Health status:	Indicates the current health status of this cluster.	Percent	**This measure is only reported at the cluster-level and not the node-level.** Every minute the cluster connects to its database and performs a simple query. If it is able to perform this operation successfully, then the value of this measure will be *Healthy*. Otherwise, the value of this measure will be *Unhealthy*. An *Unhealthy* status can occur when the cluster database is under extremely heavy load or if there is a configuration problem with a database on the cluster. The numeric values that correspond to the measure values mentioned above are as follows: **Measure Value** **Numeric Value** Healthy 1 Unhealthy 0 **Note:** This measure will report one of the **Measure Value**s listed above to indicate the current state of a cluster. In the graph of this measure however, cluster status will be indicated using the numeric equivalents only.
Is maintenance mode?:	Indicates whether/not this cluster is in the maintenance mode presently.		The values that this measure can report and their corresponding numeric values are listed in the table below:

Measurement	Description	Measurement Unit	Interpretation
			Measure Value / **Numeric Value** Yes / 1 No / 0 **Note:** • This measure will report one of the **Measure Value**s listed above to indicate whether/not a cluster is in the maintenance mode. In the graph of this measure however, the same will be indicated using the numeric equivalents only. • This measure is reported only at the cluster-level and not the node-level. • Even though your cluster might be unavailable due to maintenance tasks, the *Health status* measure of the test will report the value *Healthy* for that cluster.
Network receive throughput:	Indicates the rate at which this cluster or node receives data.	KB/Secs	For a cluster, a consistent increase in the value of these measures is indicative of excessive usage of network resources by the cluster. In such a case, compare the value of these measures across the nodes of a cluster to identify the nodes that are over-utilizing network bandwidth.
Network transmit throughput:	Indicates the rate at which this cluster or node sends data.	KB/Secs	
Disk space used:	Indicates the percentage of disk space used by this cluster/node.	Percent	If the value of this measure is close to 100% for a cluster, it indicates that the cluster is rapidly running out of storage resources. You may want to consider adding more nodes to the cluster to increase the storage space available. Alternatively, you can add fewer nodes and yet significantly increase the cluster resources by opting for node types that are by default large-sized and hence come bundled with considerable storage space.

Measurement	Description	Measurement Unit	Interpretation
			When a cluster's storage resources are rapidly depleting, you may want to compare the space usage of the nodes in cluster, so that you can quickly isolate that node that is eroding the space. Tweaking your load-balancing algorithm could go a long way in eliminating such node overloads.
Read IOPS:	Indicates the average number of disk read operations performed by this node per second.	Reads/Sec	A high value is desired for this measure, as that's the trait of a healthy node. You can compare the value of this measure across nodes to identify the node that is slowest in processing read requests.
Read latency:	Indicates the average amount of time taken by this node for disk read I/O operations.	Reads/Sec	Ideally, the value of this measure should be very low. Its good practice to compare the value of this measure across nodes of a cluster and isolate those nodes in the cluster where the value of this measure is abnormally high. Such nodes slow down I/O processing and adversely affect application performance.
Read throughput:	Indicates the average number of bytes read from disk by this node per second.	KB/Sec	A high throughput signifies faster processing of read I/O requests. A low throughput is indicative of slow read request processing. Compare the value of this measure across nodes of a cluster to isolate those nodes that have registered an abnormally low value for this measure. Such nodes not only affect cluster performance, but also the performance of dependent client applications.
Write IOPS:	Indicates the average number o disk write operations performed by this node per second.	Writes/Sec	A high value is desired for this measure, as that's the trait of a healthy node. You can compare the value of this measure across nodes to identify the node that is slowest in processing write requests.
Write latency:	Indicates the average	Secs	Ideally, the value of this measure should be

Measurement	Description	Measurement Unit	Interpretation
	amount of time taken by this node for disk write I/O operations.		very low. Its good practice to compare the value of this measure across nodes of a cluster and isolate those nodes in the cluster where the value of this measure is abnormally high. Such nodes slow down I/O processing and adversely affect application performance.
Write throughput:	Indicates the average number of bytes written to disk by this node per second.	KB/Sec	A high throughput signifies faster processing of write I/O requests. A low throughput is indicative of slow write request processing. Compare the value of this measure across nodes of a cluster to isolate those nodes that have registered an abnormally low value for this measure. Such nodes not only affect cluster performance, but also the performance of dependent client applications.

5.3.4 Elastic Block Store - EBS Test

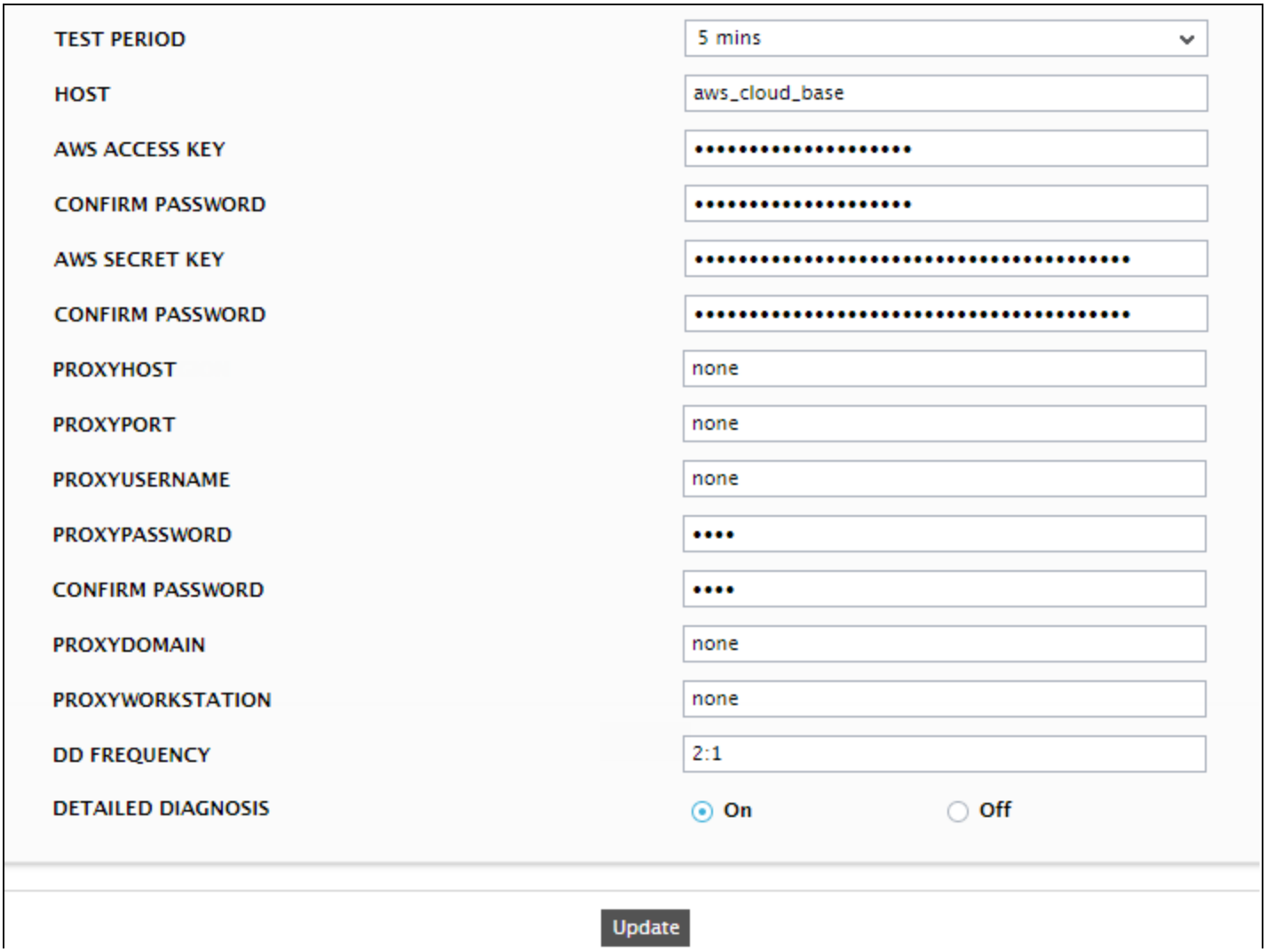
Amazon Elastic Block Store (Amazon EBS) provides persistent block level storage volumes for use with Amazon EC2 instances in the AWS Cloud. An Amazon EBS volume is a durable, block-level storage device that you can attach to a single EC2 instance. You can use EBS volumes as primary storage for data that requires frequent updates, such as system drive for an instance or storage for a database application. If such an EBS volume suddenly becomes unavailable or impaired, it is bound to adversely impact the operations of the EC2 instance attached to that volume, which in turn will damage the experience of the users of that instance. Administrators need to be promptly alerted to such problem conditions, so that they can instantly initiate remedial action and ensure high instance uptime. Besides volume status, administrators also need to track the I/O load on the EBS volume and continuously measure the ability of the volume to handle that load. This insight will enable administrators to provision the volumes with more or less I/O, so as to optimize I/O processing and maximize volume performance. The AWS Elastic Block Store - EBS test helps administrators in this exercise. The test periodically checks the health and availability status of each volume used by the EC2 instances in the monitored region and notifies administrators if any volume is in an abnormal state. Similarly, the test also tracks the I/O load on every volume and measures how well each volume processes the load - overloaded volumes and those that are experiencing processing hiccups are highlighted in the process.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for each volume of the AWS region being monitored

Test parameters:



1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided

with an "access key" and a corresponding "secret key".

- Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.

4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **PROXY HOST AND PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
7. **PROXY USERNAME AND PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none** , indicating that the proxy sever does not require authentication by default.
8. **PROXY DOMAIN AND PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
9. **DD FREQUENCY** - Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is 1:1. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against DD frequency.
10. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG system 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 against **DETAILED DIAGNOSIS**. 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 bad and normal frequencies configured for the detailed diagnosis measures should not be 0.

Measures reported by the test:

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td>State</td><td>Indicates the current state of this volume.</td><td></td><td>The values that this measure can report and their corresponding numeric values are detailed in the table below:
<table>
<tr><th>Measure Value</th><th>Description</th><th>Numeric Value</th></tr>
<tr><td>Creating</td><td>The volume is being created. The volume will be inaccessible during creation.</td><td>0</td></tr>
<tr><td>Available</td><td>The volume is available</td><td>1</td></tr>
<tr><td>In-use</td><td>The volume is in use</td><td>2</td></tr>
<tr><td>Deleting</td><td>The volume is being deleted</td><td>3</td></tr>
<tr><td>Deleted</td><td>The volume is deleted</td><td>4</td></tr>
<tr><td>Error</td><td>Some error has occurred in the volume</td><td>5</td></tr>
</table>
The detailed diagnosis of this measure will reveal when the volume was created and in which availability zone it resides.

Note:

By default, this measure will report the Measure Values listed in the table above to indicate the current availability state of a volume. In the graph of this measure however, the same will be represented using the numeric equivalents only.

If any EBS volume is found to be in an abnormal state, then you can use the detailed diagnosis of this measure to know the volume type, when that volume was created, and in which availability zone the volume resides.</td></tr>
</table>

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td>Status</td><td>Indicates the current health status of this volume</td><td></td><td>AWS EC2 periodically runs volume status checks to enable you to better understand, track, and manage potential inconsistencies in the data on an Amazon EBS volume.

Volume status checks are automated tests that run every 5 minutes and return a pass or fail status. The value that this measure reports varies with the status reported by the volume status checks. The table below describes what value this measure reports when , and also lists the numeric values that correspond to the measure values.

<table>
<tr><th>Measure Value</th><th>Description</th><th>Numeric Value</th></tr>
<tr><td>OK</td><td>If all checks pass, the status of the volume is OK.</td><td>0</td></tr>
<tr><td>Impaired</td><td>If a check fails, the status of the volume is impaired</td><td>1</td></tr>
<tr><td>Insufficient-data</td><td>If checks are in progress, then insufficient-data is reported</td><td>2</td></tr>
</table>

Note:

By default, this measure will report the Measure Values listed in the table above to indicate the current status of a volume. In the graph of this measure however, the same will be represented using the numeric equivalents only.</td></tr>
<tr><td>Idle time:</td><td>Indicates the total number of seconds</td><td>Secs</td><td></td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
	during which no read or write operations were submitted to this volume.		
Queue length:	Indicates the number of read and write operation requests waiting to be completed.	Number	A consistent increase in the value of this measure could indicate a I/O processing bottleneck on the volume.
Read operations:	Indicates the rate at which read operations were performed on this volume.	Operations/Sec	Compare the value of this measure across volumes to know which volume is too slow in processing read requests.
Write operations:	Indicates the rate at which write operations were performed on this volume.	Operations/Sec	Compare the value of this measure across volumes to know which volume is too slow in processing write requests.
Reads:	Indicates the rate at which data was read from this volume.	KB/Sec	Compare the value of this measure to identify the volume that is the slowest in responding to read requests.
Writes:	Indicates the rate at which data was written to this volume.	KB/Sec	Compare the value of this measure to identify the volume that is the slowest in responding to write requests.
Total read time:	Indicates the total time taken by all completed read operations.	Secs	A very high value for this measure could indicate that the volume took too long to service one/more read requests.
Total write time:	Indicates the total time taken by all	Secs	A very high value for this measure could indicate that the volume took too long to service

Measurement	Description	Measurement Unit	Interpretation
	completed write operations.		one/more write requests.
Provisioned IOPS (SSD)volume throughput:	Indicates the percentage of I/O operations per second (IOPS) delivered of the total IOPS provisioned for this volume.	Percent	**This measure will be reported for Provisioned IOPS volumes only.** Provisioned IOPS (SSD) volumes are designed to meet the needs of I/O-intensive workloads, particularly database workloads, that are sensitive to storage performance and consistency in random access I/O throughput. You specify an IOPS rate when you create the volume, and Amazon EBS delivers within 10 percent of the provisioned IOPS performance 99.9 percent of the time over a given year. A Provisioned IOPS (SSD) volume can range in size from 4 GiB to 16 TiB and you can provision up to 20,000 IOPS per volume. The ratio of IOPS provisioned to the volume size requested can be a maximum of 30; for example, a volume with 3,000 IOPS must be at least 100 GiB. You can stripe multiple volumes together in a RAID configuration for larger size and greater performance. For smaller I/O operations, you may even see an IOPS value that is higher than what you have provisioned - i.e., the value of this measure can be greater than 100%. This could be because the client operating system may be coalescing multiple smaller I/O operations into a smaller number of large chunks. On the other hand, if the value of this measure is consistently lower than the expected IOPS or throughput you have provisioned, then ensure that your EC2 bandwidth is not the limiting factor; your instance should be EBS-optimized (or include 10 Gigabit network connectivity) and your instance type EBS dedicated bandwidth should exceed the I/O throughput you intend to

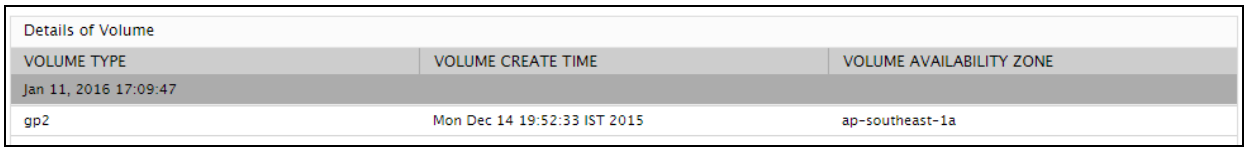
Measurement	Description	Measurement Unit	Interpretation
			drive. Another possible cause for not experiencing the expected IOPS is that you are not driving enough I/O to the EBS volumes.
Size:	Indicates the current size of this volume.	GB	For a General Purpose (SSD) Volume, volume size is what dictates the baseline performance level of the volume and how quickly it accumulates I/O credits; larger volumes have higher baseline performance levels and accumulate I/O credits faster. For a Provisioned IOPS (SSD) Volume, the ratio of IOPS provisioned to volume size can be a maximum of 30; for example, a volume with 3,000 IOPS must be at least 100 GiB. Magnetic volumes can range in size from 1 GiB to 1 TiB.
Total IOPS:	Indicates the total number of I/O operations that were performed on this volume per second.	Operations/Sec	IOPS are input/output operations per second. Amazon EBS measures each I/O operation per second (that is 256 KiB or smaller) as one IOPS. I/O operations that are larger than 256 KiB are counted in 256 KiB capacity units. For example, a single 1,024 KiB I/O operation would count as 4 IOPS; however, 1,024 I/O operations at 1 KiB each would count as 1,024 IOPS. When you create a 3,000 IOPS volume, either a 3,000 IOPS Provisioned IOPS (SSD) volume or a 1,000 GiB General Purpose (SSD) volume, and attach it to an EBS-optimized instance that can provide the necessary bandwidth, you can transfer up to 3,000 chunks of data per second (provided that the I/O does not exceed the per volume throughput limit of the volume). If your I/O chunks are very large, then the value of this measure may be lesser than what you provisioned because you are hitting the throughput limit of the volume. For example

Measurement	Description	Measurement Unit	Interpretation
			1,000 GiB General Purpose (SSD) volume has an IOPS limit of 3,000 and a volume throughput limit of 160 MiB/s. If you are using a 256 KiB I/O size, your volume will reach its throughput limit at 640 IOPS (640 x 256 KiB = 160 MiB). For smaller I/O sizes (such as 16 KiB), this same volume can sustain 3,000 IOPS because the throughput is well below 160 MiB/s. On Provisioned IOPS Volumes, for smaller I/O operations, you may even see that the value of this measure is higher than what you have provisioned. This could be because the client operating system may be coalescing multiple smaller I/O operations into a smaller number of large chunks. On the other hand, if the value of this measure is consistently lower than the expected IOPS or throughput you have provisioned for a Provisioned IOPS volume, then ensure that your EC2 bandwidth is not the limiting factor; your instance should be EBS-optimized (or include 10 Gigabit network connectivity) and your instance type EBS dedicated bandwidth should exceed the I/O throughput you intend to drive. Another possible cause for not experiencing the expected IOPS is that you are not driving enough I/O to the EBS volumes. Magnetic volumes deliver approximately 100 IOPS on average, with burst capability of up to hundreds of IOPS.
IOPS limits:	Indicates the IOPS limit of this volume.	Operations/Sec	For Provisioned IOPS volumes, the IOPS limit is specified when creating the volumes. For General Purpose IOPS volumes, the volume size dictates the baseline IOPS limit of that volume and how quickly it accumulates I/O credits.

Measurement	Description	Measurement Unit	Interpretation
IOPS utilization:	Indicates the percentage of provisioned IOPS or IOPS limit that is being utilized by this volume.	Percent	This metric can also help you identify over-utilized volumes, which could be impacting application performance. In these cases, you could improve performance by upgrading to a different volume type or provisioning more IOPS.
Throughput:	Indicates the rate of reads and writes processed by this volume.	KB/Second	A consistent drop in this value could indicate a I/O processing bottleneck on the volume. You may want to closely track the variations to this measure, so that you can proactively identify the volume that may soon reach its throughput limit. The maximum throughput of each volume type is indicated below: • **General purpose volumes - 160 MiB/sec** • **Provisioned IOPS volumes - 320 MiB/sec** • **Magnetic volumes - 40- 90 MiB/sec** If your I/O chunks are very large, then a volume will reach its throughput limit much before its IOPS limit is reached. If you are not experiencing the throughput you have provisioned, ensure that your EC2 bandwidth is not the limiting factor; your instance should be EBS-optimized (or include 10 Gigabit network connectivity) and your instance type EBS dedicated bandwidth should exceed the I/O throughput you intend to drive.

Detailed Diagnosis:

The detailed diagnosis of the **State** measure of a volume will reveal when the volume was created and in which availability zone it resides.



Details of Volume

VOLUME TYPE	VOLUME CREATE TIME	VOLUME AVAILABILITY ZONE
Jan 11, 2016 17:09:47		
gp2	Mon Dec 14 19:52:33 IST 2015	ap-southeast-1a

Figure 5.16: The detailed diagnosis of the State measure of the AWS Elastic Block Store - EBS Test

5.3.5 Simple Email Service - SES Test

Amazon Simple Email Service (Amazon SES) is a cost-effective email service built on the reliable and scalable infrastructure that Amazon.com developed to serve its own customer base. This service allows you to build an email functionality into an application that you are running on AWS. With Amazon SES, you can send transactional email, marketing messages, or any other type of high-quality content to your customers. You can also use Amazon SES to receive messages and deliver them to an Amazon S3 bucket, call your custom code via an AWS Lambda function, or publish notifications to Amazon SNS.

Amazon SES has a set of sending limits to regulate the number of email messages that you can send and the rate at which you can send them. Depending upon the level of email activity in your environment, you may want to modify these limits, as any violation will result in mails not being sent at all. You may hence have to closely study the email activity in your environment and determine whether/not the sending limits need to be fine-tuned. The **Simple Email Service - SES** test helps with this! By reporting the send quotas configured along with the count of mails sent and the send rate for the monitored AWS region, this test readily provides you with all the information you need to take the right decision with regards to whether/not the quota needs to be reset.

Also, the key measure of the performance of any email service is successful message delivery. If a majority of the delivery attempts made at any given point in time resulted in bounces, rejections, or complaints, it is a problem condition that warrants an investigation. The **Simple Email Service - SES** test proactively alerts you to such abnormalities! For the monitored region, the test reports the count and percentage of emails bounced, mails rejected, and complaints received, and notifies you if these values exceed acceptable limits.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for the region being monitored

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured

3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
7. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
8. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to none.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Sending quota:	Indicates the maximum number of emails that can be sent in a day.	Emails/Day	The sending quota reflects a rolling time period. Every time you try to send an email, Amazon SES checks how many emails you sent in the previous 24 hours. As long as the total number of emails that you have sent is less than

Measurement	Description	Measurement Unit	Interpretation
			your quota, your send request will be accepted and your email will be sent. If you have already sent your full quota, your send request will be rejected with a throttling exception. You will not be able to send more emails until some of the previous sending rolls out of its 24-hour window.
Total sent:	Indicates the total number of emails sent during the last 24 hours.	Number	If the value of this measure keeps growing closer to the value of the value of the Sending quota measure, it implies a high level of email activity in the region. Under such circumstances, it is best to increase the sending quota, so that the quota is not violated, causing SES to stop sending emails.
Sents:	Indicates the percentage of sending quota that was exhausted in the last 24 hours.	Percent	This measure is computed using the following formula: (Total sent / Sending quota) * 100 If the value of this measure is consistently higher than 50%, it implies a high level of email activity in the region. Under such circumstances, it is best to increase the sending quota, so that the quota is not violated, causing SES to stop sending emails.
Max send rate:	Indicates the maximum number of emails that can send per second.	Emails/Sec	You can exceed this limit for short bursts, but not for a sustained period of time.
Current bounces:	Indicates the number of emails that were bounced during the last measurement period.	Number	An email is hard-bounced when the email is rejected by the recipient's ISP or rejected by Amazon SES because the email address is on the Amazon SES suppression list. This measure reports the count of hard bounces

Measurement	Description	Measurement Unit	Interpretation
			alone. The value of this measure should be kept at a minimum, as excessive bounces constitute abuse and can put your AWS account at the risk of termination.
Bounce:	Indicates the percentage of emails that were bounced during the last measure period.	Percent	Ideally, the value of this measure should be very low. A high value constitutes abuse and can put your AWS account at the risk of termination.
Complaints:	Indicates the number of complaints received during the last measure period.	Number	If an email is accepted by the ISP and delivered to the recipient, but the recipient does not want the email and clicks a button such as "Mark as spam.", then SES will send you a complaint notification. The value of this measure should be kept at a minimum, as a large number of complaints constitute abuse and can put your AWS account at the risk of termination.
Complaint:	Indicates the percentage of complaints received by this region during the last measure period.	Percent	Ideally, the value of this measure should be very low. A high value constitutes abuse and can put your AWS account at the risk of termination.
Current rejected:	Indicates the number of emails that were rejected during the last measurement period.	Number	A rejected email is an email that Amazon SES initially accepted, but later rejected because the email contained a virus. Amazon SES notifies you by email and does not send the message. A high value for this measure is a cause for concern as it could indicate that your email system is severely infected.

Measurement	Description	Measurement Unit	Interpretation
Rejected:	Indicates the percentage of emails that were rejected during the last measurement period.	Percent	A high value for this measure is a cause for concern as it could indicate that your email system is severely infected.
Current delivery attempts:	Indicates the number of mails sent during the last measurement period.	Number	

5.3.6 Relational Database Service - RDS Test

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizeable capacity for an industry-standard relational database and manages common database administration tasks. It also manages backups, software patching, automatic failure detection, and recovery.

The basic building block of Amazon RDS is the DB instance. A DB instance is an isolated database environment in the cloud. A DB instance can contain multiple user-created databases, and you can access it by using the same tools and applications that you use with a stand-alone database instance.

Each DB instance runs a DB engine. Amazon RDS currently supports the MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server DB engines.

The computation and memory capacity of a DB instance is determined by its DB instance class. For each DB instance, you can select from 5 GB to 6 TB of associated storage capacity. Each DB instance class has minimum and maximum storage requirements for the DB instances that are created from it. You can select the DB instance that best meets your needs. If your needs change over time, you can change DB instances. For example, initially, you may have launched a standard (previous generation) DB instance, which provides a balance of compute, memory, and network resources for your applications. However later, based on usage, you may have realized that a burst capable - current generation DB instance, with the capability to burst to full CPU usage, is ideal for your needs. In such circumstances, RDS facilitates the switch from one type DB instance to another. But to understand which DB instance class best suits your needs and make timely and accurate adjustments to your DB instance class selection, you will have to constantly track the CPU, memory, network, and space usage of each active DB instance on the cloud and derive usage patterns. Also, to ensure optimal storage performance, you additionally need to keep an eye on the I/O operations performed on the DB instances and identify latent DB instances. This is exactly what the Relational Database Service - RDS enables you to achieve.

This test closely tracks the current status, resource usage, and I/O activity of every active DB instance on a monitored region, and brings the following to light:

- Is any DB instance in an abnormal state presently?
- How are the DB instances using the CPU resources they have been configured with? Is any DB instance consuming high levels of CPU consistently? Should the DB instance class be changed?
- Does the DB instance have enough RAM? Will changing the DB instance class help in reducing the memory pressure on the instance?
- Do any db.t2 instances have a poor CPU credit balance?
- Is the disk I/O queue of any DB instance abnormally high? Which instance is this and when is I/O latency on that instance very high - when reading from or writing to the instance?
- Which DB instance is hungry for network bandwidth?
- Do all DB instances have enough free space? If not, which ones are rapidly running short of space?

Target of the test : Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for each active DB instance / DB instance class / DB engine name (depending upon the option you choose from the **RDS FILTER** drop-down) in the monitored region

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST** – The host for which the test is being configured
3. **AWS ACCESS KEY** - To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD** - Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics

collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.

7. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
8. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.
9. **RDS FILTER** - By default, this test reports metrics for each active DB instance on the cloud. This is why, this flag is set to **DBInstanceIdentifier**, by default. If needed, you can pick either of the following options from this drop-down:
   - **DatabaseClass:** The computation and memory capacity of a DB instance is determined by its DB instance class. If you select this option, then this test will report metrics for each DB instance class. In other words, eG will aggregate metrics for all databases that belong to a DB intance class, and will present these metrics at the macro class-level.
   - **EngineName:** Each DB instance runs a DB engine. Amazon RDS currently supports the MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server DB engines. Each DB engine has its own supported features, and each version of a DB engine may include specific features. If you select this option, then this test will report metrics for every DB engine. In this case, eG will aggregate metrics for all databases using a particular engine, and will present these metrics at the macro engine-level.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
RDS instance status:	Indicates the current status of this DB instance.		**This measure is reported only for a DB instance descriptor.** The values that this measure reports, the significance of each of these values, and the numeric values that correspond to them are discussed in the table below: **Measure Value** / **Description** / **Numeric Value** Failed / The instance has failed and Amazon RDS / 0

Measurement	Description	Measurement Unit	Interpretation
			was unable to recover it. Perform a point- in- time restore to the latest restorable time of the instance to recover the data. Available The instance is healthy and available 1 Backing-up The instance is currently being backed up. 2 Creating The instance is being created. The instance is inaccessible while it is being created. 3 Inaccessible-encryption-credentials The KMS key used to encrypt or decrypt the DB instance could not be accessed. 4 Incompatible-credentials The supplied CloudHSM username or password is incorrect. Please update the CloudHSM credentials for 5

Measurement	Description	Measurement Unit	Interpretation
			the DB instance. Incompatible-network: Amazon RDS is attempting to perform a recovery action on an instance but is unable to do so because the VPC is in a state that is preventing the action from being completed. This status can occur if, for example, all available IP addresses in a subnet were in use and Amazon RDS was unable to get an IP address for the DB instance. 6 Incompatible-option-group: Amazon RDS attempted to apply an option group change but was unable to do so, and Amazon RDS was unable to roll back to the 7

Measurement	Description	Measurement Unit	Interpretation
			previous option group state. Consult the Recent Events list for the DB instance for more information. This status can occur if, for example, the option group contains an option such as TDE and the DB instance does not contain encrypted information.
			Incompatible-parameters: Amazon RDS 8 was unable to start up the DB instance because the parameters specified in the instance's DB parameter group were not compatible. Revert the parameter changes or make them compatible with the

Measurement	Description	Measurement Unit	Interpretation
			instance to regain access to your instance. Consult the Recent Events list for the DB instance for more information about the incompatible parameters. Incompatible-restore: Amazon RDS is unable to do a point-in-time restore. Common causes for this status include using temp tables or using MyISAM tables. 9 Maintenance: Amazon RDS is applying a maintenance update to the DB instance. 10 Modifying: The instance is being modified because of a customer request to modify the instance. 11

Measurement	Description	Measurement Unit	Interpretation		
			Rebooting	The instance is being rebooted because of a customer request or an Amazon RDS process that requires the rebooting of the instance.	12
			Renaming	The instance is being renamed because of a customer request to rename it.	13
			Resetting-master-credentials	The master credentials for the instance are being reset because of a customer request to reset them.	14
			Restore-error	The DB instance encountered an error attempting to restore to a point-in-time or from a snapshot.	15
			Upgrading	The database engine version is being upgraded.	16

Measurement	Description	Measurement Unit	Interpretation
			Storage-full 17 — The instance has reached its storage capacity allocation. This is a critical status and should be remedied immediately; you should scale up your storage by modifying the DB instance. Set alarms to warn you when storage space is getting low so you don't run into this situation. Deleting 18 — The instance is being deleted. **Note:** This measure reports the Measure Values listed in the table above to indicate the current status of a DB instance. In the graph of this measure however, the same will be represented using the corresponding numeric equivalents only.
CPU credit usage:	Indicates the number of CPU units consumed by this T2 DB instance/ all DB instances that	Number	These measures are reported only for individual T2 instances, instances that belong to T2 DB instance classes, and the DB engines used only by T2 instances. A CPU Credit provides the performance of a full

Measurement	Description	Measurement Unit	Interpretation
	belong to this T2 DB instance class / all T2 DB instances using this DB engine, during the last measurement period.		CPU core for one minute. Traditionalinstance types provide fixed performance, while T2 instances provide a baseline level of CPU performance with the ability to burst above that baseline level. The baseline performance and ability to burst are governed by CPU credits. One CPU credit is equal to one vCPU running at 100% utilization for one minute. Other combinations of vCPUs, utilization, and time are also equal to one CPU credit; for example, one vCPU running at 50% utilization for two minutes or two vCPUs running at 25% utilization for two minutes. Each T2 instance starts with a healthy initial CPU credit balance and then continuously (at a millisecond-level resolution) receives a set rate of CPU credits per hour, depending on instance size. When a T2 instance uses fewer CPU resources than its base performance level allows (such as when it is idle), the unused CPU credits (or the difference between what was earned and what was spent) are stored in the credit balance for up to 24 hours, building CPU credits for bursting. When your T2 instance requires more CPU resources than its base performance level allows, it uses credits from the CPU credit balance to burst up to 100% utilization. The more credits your T2 instance has for CPU resources, the more time it can burst beyond its base performance level when more performance is needed. This implies that ideally, the value of the CPU credit usage measure should be low for an instance and the value of the CPU credit balance for that instance should be high, as that way, an instance is assured of more CPU resources when performance demands increase. By comparing the value of this

Measurement	Description	Measurement Unit	Interpretation
			measure across instances, you can precisely identify the instance that has used up a sizeable portion of its CPU credits.
CPU credit balance:	Indicates the number of CPU credits that an instance has accumulated.	Number	
CPU utilization:	Indicates the percentage of CPU utilized by this DB instance / DB instance class / DB engine	Percent	A value close to 100% for this measure for any DB instance is indicative of excessive CPU usage by that instance. Track the variations to the value of this measure for such an instance closely, and figure out whether CPU usage is consistently high and close to 100%. If so , you can conclude that the instance requires more CPU than what's been allocated to it. You may want to change to the DB instance class definition to allot more CPU resources to all instances it governs.
Binlog disk usage:	Indicates the amount of disk space occupied by binary logs on this DB instance / all DB instances of this DB instance class / all DB instances using this DB engine	KB	The binary log on MySQL has two important purposes: • **For replication, the binary log on a master replication server provides a record of the data changes to be sent to slave servers. The master server sends the events contained in its binary log to its slaves, which execute those events to make the same data changes that were made on the master. .** • **Certain data recovery operations require use of the binary log. After a backup has been restored, the events in the binary log that were recorded after the backup was made are re-executed. These events bring databases up to date from the**

Measurement	Description	Measurement Unit	Interpretation
			point of the backup. Typically, MySQL uses several logging formats to record information in the binary log. There are three logging formats: • Replication capabilities in MySQL originally were based on propagation of SQL statements from master to slave. This is called statement-based logging. • In row-based logging, the master writes events to the binary log that indicate how individual table rows are affected. • A third option is also available: mixed logging. With mixed logging, statement- based logging is used by default, but the logging mode switches automatically to row- based in certain cases. MySQL on Amazon RDS supports both the row-based and mixed binary logging formats for MySQL version 5.6. The default binary logging format is mixed. For DB instances running MySQL versions 5.1 and 5.5, only mixed binary logging is supported. If the value of this measure grows consistently, it could mean that large binary files are being created. At this juncture, you may want to check the logging format configured for MySQL on Amazon RDS. This is because, very often, row-based binary logging format can result in very large binary log files. If you do not change the logging mode, then such files will continue to be created, thereby reducing the amount of storage space available for a DB instance. This in turn can increase the amount of time to perform a

Measurement	Description	Measurement Unit	Interpretation
			restore operation of a DB instance.
Database connections:	Indicates the number of database connections currently used by this instance / all instances that belong to this DB instance class / all instances using this DB engine	Number	
Disk queue depth:	Indicates the number of outstanding IOs (read/write requests) waiting to access this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine.	Number	If the value of this measure keeps increasing steadily and significantly for a DB instance, it could indicate that the DB instance is latent, and is unable to process I/O requests quickly. The value of this measure therefore should be low at all times.
Freeable memory:	Indicates the amount of available random access memory for this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine	MB	A high value is desired for this measure to ensure peak performance of a DB instance.
Replica lag time:	Indicates the amount of time a Read Replica DB Instance lags behind this source DB Instance / all source	Secs	**This measure applies to MySQL read replicas only.** f your system runs on Amazon Relational Database Service (RDS) you may have opted to configure one or more replicas for your main MySQL database(s). This means you have a

Measurement	Description	Measurement Unit	Interpretation
	DB instances that belong to this DB instance class / all source DB instances using DB engine		master RDS instance and at least one slave RDS instance which receives updates from the master. This process is called replication. Replication ensures that changes made on the master database also happen on the slave after some period of time. For a variety of reasons this period of time can increase. For example, a long-running query or erroneous query can cause replication to slow down or stop entirely. This results in replication lag: changes made on your main database aren't showing up on the slave replica because the replica is lagging behind. If the value of this measure is increasing consistently for a DB instance, it is a cause for concern, as it indicates that the slave is not in sync with the master and will take a long time to catch up. If for any reason the master DB instance fails at this juncture, there is bound to be significant data loss owing to the master-slave non-sync. When there is a replication issue the output of *show slave status*; is quite useful in debugging and resolving it. You need to review the values of: Slave_SQL_Running Last_Error Last_SQL_Error When a particular SQL query failed on the slave it could be that execution of queries in general has stopped. This is indicated by *Slave_SQL_Running* having the value *No*. In that case you'll either need to: • Remedy the error by fixing the issue that caused the SQL query to fail. • Decide to resume replication by letting the

Measurement	Description	Measurement Unit	Interpretation
			slave ignore that error. The former situation can be tricky as it requires you to figure out what data or query is problematic based on the values of *Last_Error* and *Last_SQL_Error*. These fields may provide enough information to determine any incorrect records but this is not always the case. In the latter case you would execute the following command on the slave: *CALL mysql.rds_skip_repl_error;* You should only run this command when you've determined that skipping the SQL query won't lead to inconsistent data or incorrect data on the slave (or, at least, that this is allowed to occur by skipping that particular SQL query).
Swap usage:	Indicates the amount of swap space used on this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine.	KB	
Read IOPS:	Indicates the rate at which disk read I/O operations were performed by this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine	Reads/Sec	Ideally, the value of this measure should be high. A consistent drop in this value could indicate a read latency.
Write IOPS:	Indicates the rate at	Writes/Sec	Ideally, the value of this measure should be high.

Measurement	Description	Measurement Unit	Interpretation
	which disk write I/O operations were performed by this DB instance / all DB instances that belong to this DB instance class / all DB instances using this DB engine		A consistent drop in this value could indicate a write latency.
Read latency:	Indicates the average amount of time this DB instance / all DB instances of this instance class / all DB instances using this engine, took to service read requests.	Secs	Ideally, the value of this measure should be low. A consistent rise in this value could indicate a read latency. Compare the value of this measure across DB instances to know which instance is the slowest in servicing read requests.
Write latency:	Indicates the average amount of time this DB instance / all DB instances of this instance class / all DB instances using this engine, took to service write requests.	Secs	Ideally, the value of this measure should be low. A consistent rise in this value could indicate a write latency. Compare the value of this measure across DB instances to know which instance is the slowest in servicing write requests.
Read throughput:	Indicates the rate at which data was read from the disk by this DB instance / all DB instances of this instance class / all DB instances using this DB engine.	KB/Sec	Ideally, the value of this measure should be high. A steady decrease in this value could indicate a read latency. Compare the value of this measure across DB instances to know which instance is the slowest in servicing read requests.
Write throughput:	Indicates the rate at	KB/Sec	Ideally, the value of this measure should be high.

Measurement	Description	Measurement Unit	Interpretation
	which data was written to the disk by this DB instance / all DB instances of this instance class / all DB instances using this DB engine.		A steady decrease in this value could indicate a write latency. Compare the value of this measure across DB instances to know which instance is the slowest in servicing write requests.
Network receive throughput:	Indicates the incoming network traffic on this DB instance / all DB instances that belong to this instance class / all DB instances using this engine.	KB/Secs	The value of these measures includes both customer database traffic and Amazon RDS traffic used for monitoring and replication. A high value for these measures is indicative of high bandwidth usage by a DB instance. Under such circumstances, compare the value of the Network receive throughput measure with that of th Network transmit throughput measure to determine when the maximum bandwidth was consumed - when sending data or when receiving it?
Network transmit throughput:	Indicates the outgoing network traffic on this DB instance / all DB instances that belong to this instance class / all DB instances using this engine.	KB/Secs	
Total storage space:	Indicates the total amount of storage space allocated to this DB instance / all DB instances that belong to this instance class / all DB instances using this DB engine.	MB	
Used storage space:	Indicates the amount of storage space used by this DB instance / all DB instances that	MB	Compare the value of this measre across DB instances to know which instance is consuming storage space excessively.

Measurement	Description	Measurement Unit	Interpretation
	belong to this instance class / all DB instances using this DB engine.		
Free storage space:	Indicates the amount of storage space still unused by this DB instance / all DB instances that belong to this instance class / all DB instances using this DB engine.	MB	A high value for this measure is ideal. Compare the value of this measure across DB instances to know which instance is left with very little free space.
Free storage space:	Indicates the percentage of storage space allocated to this DB instance / all DB instances that belong to this instance class / all DB instances using this DB engine, which is still available for use.	Percent	A value close to 100% is desired for this measure. If the value of this measure is below 50% consistently, it indicates that the DB instance is not sized with adequate resources. You may want to consider changing the DB instance class of that instance, so that more storage resources are available to it.

5.3.7 Billing Test

AWS Billing and Cost Management is the service that you use to pay your AWS bill, monitor your usage, and budget your costs.

When budgeting costs, this service also provides forecasts of your estimated costs. Using the **Billing test** you can configure thresholds for this estimate for each service you subscribe to and also for a roll-up of estimated charges of all services. The test will then proactively alert you if the estimate is about to exceed your budget, and thus enable you to initiate measures for avoiding cost overruns.

Note:

This test will run for the 'us-east' region only. Since this region stores Amazon CloudWatch metrics for worldwide estimated charges, the *Estimated charges* that this region reports per service will be the consolidated charges for all regions that use that particular service.

Target of the test: Amazon EC2 Region

Agent deploying the test: A remote agent

Output of the test: One set of results for each service subscribed

Test parameters:

1. **TEST PERIOD** - How often should the test be executed
2. **HOST**– The host for which the test is being configured
3. **AWS ACCESS KEY**- To monitor an AWS EC2, the eG agent has to be configured with the "access key" of a user with a valid AWS account. To obtain the access key, follow the steps given below:
   - Sign up for a new AWS account from the http://aws.amazon.com/ec2/ page.
   - Provide the details of the user for whom you wish to create the AWS account.
   - Based on the AWS EC2 Regions, you will be requested to choose the pricing for an instance to be deployed in the AWS EC2.
   - Once the payment is made, the user will be automatically signed in to the AWS account.
   - From the newly created AWS account, you can request for an "access key". You will be provided with an "access key" and a corresponding "secret key".
   - Provide the access key in the AWS ACCESS KEY text box; this will enable the eG agent to communicate with the AWS API and collect the required metrics.
4. **CONFIRM PASSWORD**- Confirm the password by retyping it here.
5. **AWS SECRET KEY**- Provide the secret key corresponding to the access key that you had obtained through your AWS account.
6. **PROXYHOST** and **PROXY PORT**– In some environments, all communication with the AWS EC2 cloud and its regions could be routed through a proxy server. In such environments, you should make sure that the eG agent connects to the cloud via the proxy server and collects metrics. To enable metrics collection via a proxy, specify the IP address of the proxy server and the port at which the server listens against the **PROXY HOST** and **PROXY PORT** parameters. By default, these parameters are set to **none** , indicating that the eG agent is not configured to communicate via a proxy, by default.
7. **PROXY USERNAME** and **PROXY PASSWORD** - If the proxy server requires authentication, then, specify a valid proxy user name and password in the **PROXY USER NAME** and **PROXY PASSWORD** parameters, respectively. By default, these parameters are set to **none**, indicating that the proxy sever does not require authentication by default.
8. **PROXY DOMAIN** and **PROXY WORKSTATION**- If a Windows NTLM proxy is to be configured for use, then additionally, you will have to configure the Windows domain name and the Windows workstation name required for the same against the **PROXY DOMAIN** and **PROXY WORKSTATION** parameters. If the environment does not support a Windows NTLM proxy, set these parameters to **none**.

Measures reported by the test:

Measurement	Description	Measurement Unit	Interpretation
Estimated charges:	Indicates the estimated cost of this service for all regions using the service.	USD	Compare the value of this measure across services to know which service you will be spending the most on in the future. You can be notified if cost estimations for a service exceed an acceptable limit, by configuring such a limit as a the maximum threshold for this measure for that service. Based on these alarms, you can set out to change how frequently you actually use that service, so as to reduce related overheads. For the **Total** descriptor, this measure will report the total estimated charges across all services.

6

Conclusion

This document has clearly explained how eG Enterprise monitors the **AWS EC2 cloud and region**. For more information on eG Enterprise, please visit our web site at www.eginnovations.com or write to us at sales@eginnovations.com.