Track server errors with custom metrics and alerts

Published: 2020-02-22

While ExtraHop offers over 4,500 built-in metrics, there are many situations in which it is more effective to track network issues with a custom metric. For example, while built-in metrics show you issues with HTTP responses and requests, a custom metric can identify 500-level server errors. These types of errors can indicate gateway issues, an overloaded server, or configuration issues.

In this walkthrough, you will learn how to write a trigger to collect custom metrics for server errors and how to create an alert that only sends an email notification when those specific errors occur. Then, you will be able to answer the following types of questions about server errors on your network:

- Are my customers receiving 500-level server errors?
- Which error codes occurred?
- When did the errors occur?
- What URI was the customer attempting to access?
- What is the IP address of the client and server affected in the transaction?

Prerequisites

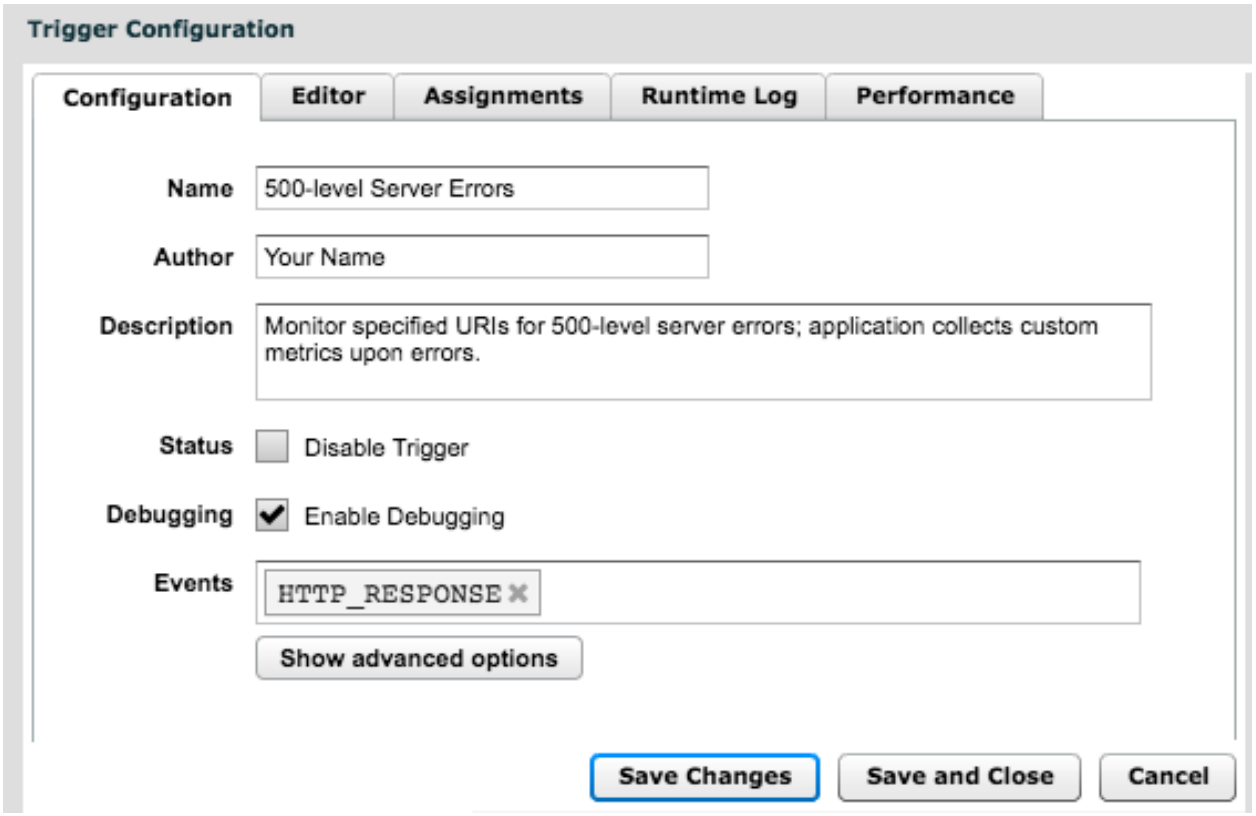
- You must have access to an ExtraHop Discover appliance with a user account that has unlimited privileges.
- Your Discover appliance must have network data with web server traffic.
- Your Discover appliance must be configured to send email notifications before you can send alert emails.
- Familiarize yourself with the concepts in this walkthrough by reading Triggers and Alerts.
- Familiarize yourself with the processes of creating triggers by reading Build a trigger.
- It is helpful to have basic JavaScript knowledge.

Write a trigger to collect error data

First, let's create a trigger that monitors certain URIs for 500-level server errors. When errors occur, the trigger collects data such as error codes and server and client IP addresses and commits that data as custom metrics to an application.

1. Log into the Web UI of a Discover appliance.
2. Click the System Settings icon, and then click **Triggers**.
3. Click **New** to open the Trigger Configuration window.
4. In the **Name** field, type a name for the trigger. For this walkthrough, type `500-level Server Errors`.
5. In the **Description** field, type information about the trigger. For this walkthrough, type `Monitor specified URIs for 500-level server errors; application collects custom metrics upon errors`.
6. Click **Enable Debugging**.
7. Click in the **Events** field and select **HTTP_RESPONSE** from the list.

   The following figure displays the trigger settings we configured above:



8. Click the **Editor** tab.
9. In the Trigger Script editor, copy and paste the following code:

```
// VARIABLES TO EDIT //

// Edit this array with hosts and URIs to monitor
var CHECK_URI_LIST = [
'www.example.com',
'example.com/about/main_page',
'http://www.example.com/about/main_page',
];

// Edit this array with client IP addresses to ignore
var IGNORE_CLIENT_IP = [
'180.57.175.147',
'172.16.156.130'
];

// END VARIABLES TO EDIT //

// DO NOT EDIT REMAINDER OF SCRIPT //

// If client IP is in array above, end process
if (IGNORE_CLIENT_IP.indexOf(Flow.client.ipaddr.toString()) > -1){
//debug ('Ignoring client IP: ' + Flow.client.ipaddr);
return}

// If URI is empty, end process
var uri = HTTP.uri || HTTP.host;
if (uri === null){//debug ('No URI Found, ending');
return}
```

```
// If URI/hostname does not match array above, end process
var matches = CHECK_URI_LIST.filter(condition => uri.indexOf(condition) >
 -1);
if (matches.length === 0) {
//debug ('URI or host did not match: ' + uri);
return}

var app = "HTTP Server Errors",
code = HTTP.statusCode,
client = Flow.client.ipaddr.toString(),
server = Flow.server.ipaddr.toString(),
detail = 'Code: ' + code ' || Server: ' + server + ' || Client: ' +
 client + ' || URI: '
          + uri;

if (code < 500 || code > 600) {return}

var code = HTTP.statusCode.toString();

// If URI matches and is a 5xx error, commit custom metrics
// to the application, which is created upon the initial event
Application(app).metricAddCount('HTTP_error', 1);
Application(app).metricAddDetailCount('HTTP_error_uri', uri, 1);
Application(app).metricAddDetailCount('HTTP_error_serverIP', server, 1);
Application(app).metricAddDetailCount('HTTP_error_clientIP', client, 1);
Application(app).metricAddDetailCount('HTTP_error_allDetail', detail, 1);
debug ('Detail: ' + detail);
```

10. In the `CHECK_URI_LIST` variable array, replace example.com with the hosts, URIs, and host-URI combinations that you want the trigger to monitor.
11. In the `IGNORE_CLIENT_IP` variable array, replace the example IP addresses (180.57.175.147 and 172.16.156.130) with the IP addresses that you want the trigger to ignore.

    Delete or comment out this array if there are no IP addresses to ignore.
12. Click **Save and Close**.
    The trigger editor validates the syntax of your script. Invalid actions, syntax errors, or deprecated elements will prevent you from saving the trigger until you fix the code or disable syntax validation.

After the trigger is assigned and running, it creates the following components that we will reference when configuring alert settings in the next sections:

HTTP Server Errors

The trigger commits data collected from the custom metrics to this application.

HTTP_error

A custom count metric that collects the number of 500-level errors that occur.

HTTP_error_allDetail

A custom detail metric that collects the code number, URI, server IP and client IP on which each error occurred.

Assign the trigger to a device

Before the trigger can run, it must be assigned to at least one device. In this step, we will assign the trigger to one or more HTTP servers that support traffic over the URIs you specified in the trigger.

1. Click **Metrics** from the top menu.

   The Activity Groups page appears, which lists the number of devices for each protocol.
2. Click **HTTP Servers**, and then click **Devices** from the left pane.
3. From the device list, select the checkbox next to one or more devices that support traffic over the URIs.

4. At the top of the page, click **Assign Trigger** to open a list of triggers.
5. Select the trigger named **500-level Server Errors** that we created in the previous section, and then click **Assign Triggers**.

Next steps

Tip: Assign triggers only to relevant devices to avoid unnecessary performance impact on the system. A good way to ensure that a trigger runs only on relevant devices is to create a device group and assign the trigger to that group.

Configure an alert to track a custom metric

Next, let's configure alert settings that will issue an alert and send an email notification each time a 500-level error occurs on the URIs watched by the trigger.

In the alert settings, we will reference the following custom metrics that we created in the trigger script:

HTTP_error

The custom count metric that collects the number of 500-level errors that occur. We will configure alert settings to track this metric and issue an alert each time an error occurs.

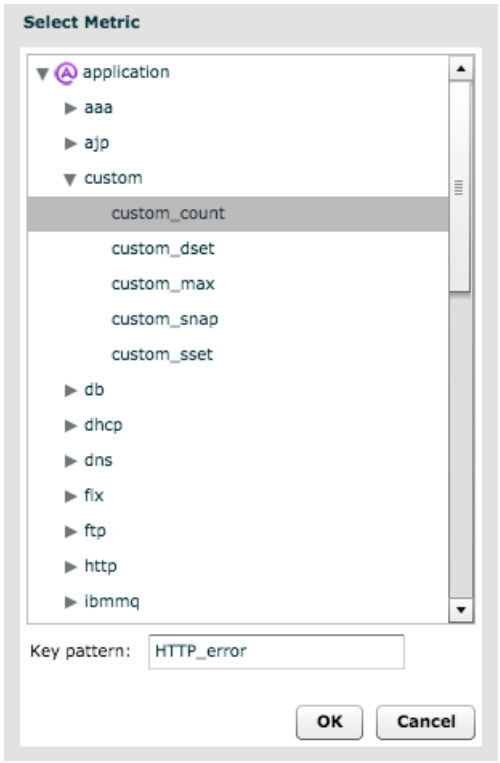
HTTP_error_allDetail

The custom detail metric that collects the code number, URI, server IP address, and client IP address on which each error occurred. We will configure alert settings to display these error details in alert emails.

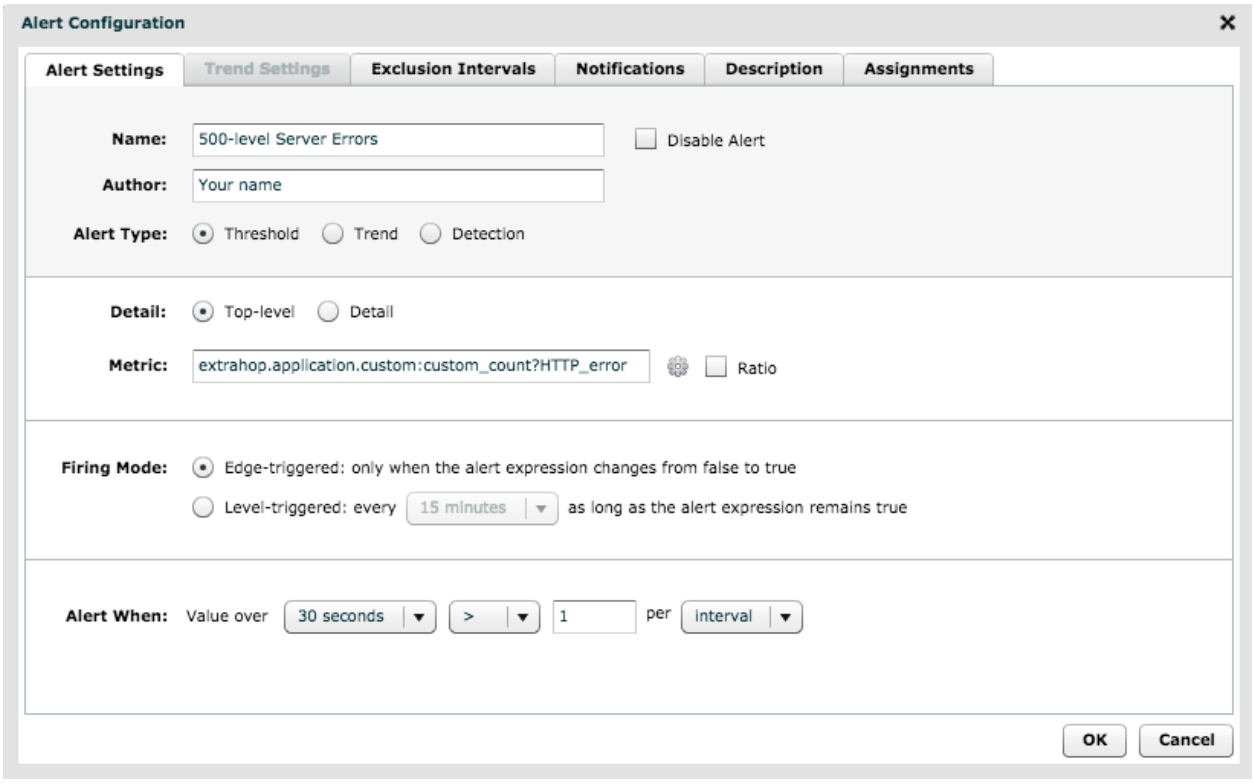
Before you begin

The Discover appliance must be configured for email notifications.

1. Click the System Settings icon and then click **Alerts**.
2. Click **New** and then type a name for the alert in the **Name** field. For this walkthrough, type `500-level Server Errors`.
3. In the Alert Type section, select **Threshold** to issue an alert when the tracked metric event occurs.
4. Complete the following steps to specify the metric that the alert will track, which is the custom count metric named HTTP_error.
   a) In the Detail section, select **Top-level** and then click the Select metric icon.
   b) In the Select Metric window, click **application** > **custom** > **custom_count**.
   c) In the **Key pattern** field, type `HTTP_error` and then click **OK**.



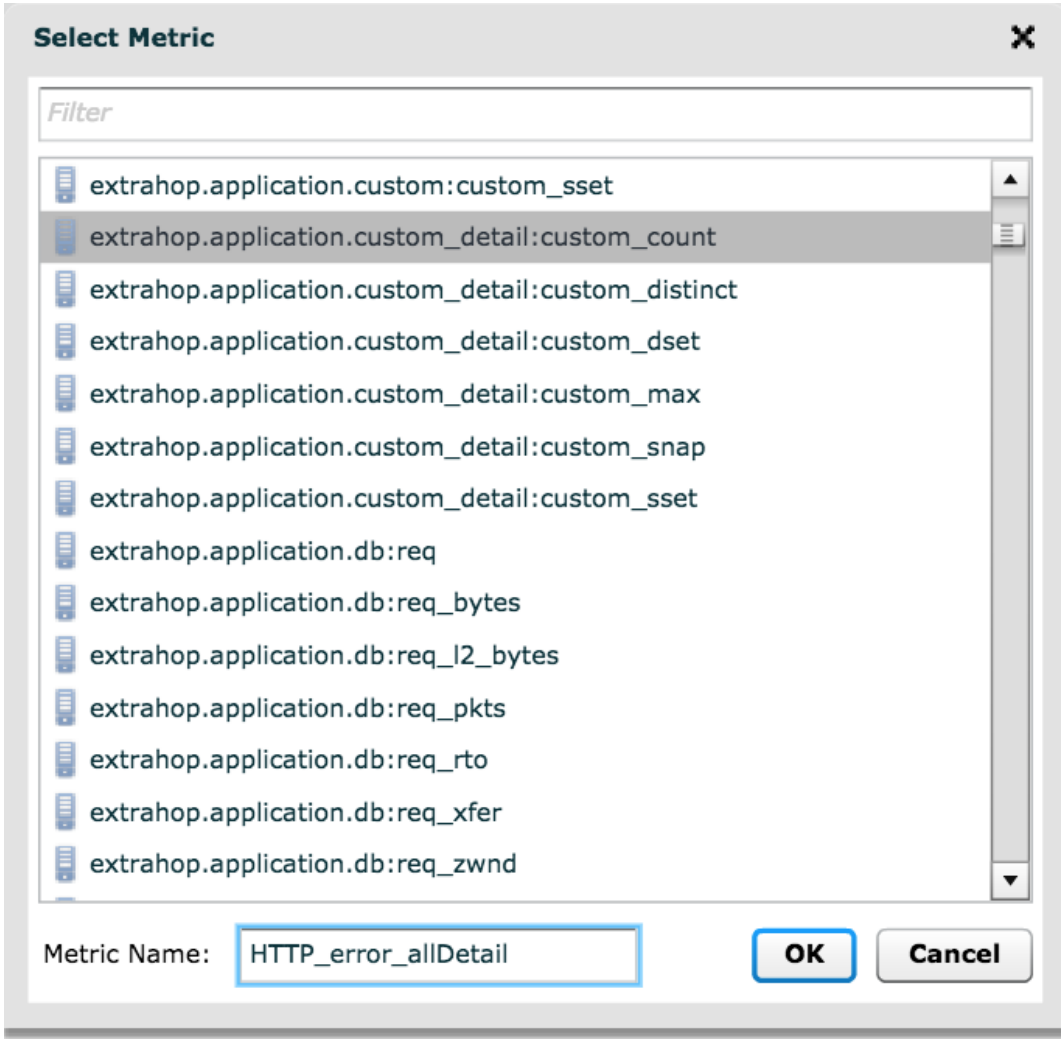
5. In the Firing Mode section, select **Edge-Triggered** to generate an alert for each occurrence of the tracked metric.
6. In the Alert When section, specify the following expression to generate an alert if the tracked metric event occurs more than once in a 30-second time period: `Value over 30 seconds > 1`
The following figure displays the alert email notification settings we configured above:



7. Click the **Description** tab and then type a description of the alert. For this walkthrough, type `Alert issued when a 500-level server error occurs on monitored URIs.`
8. Click the **Notifications** tab.
9. In the Additional email addresses section, type an email address that should receive alert notifications.

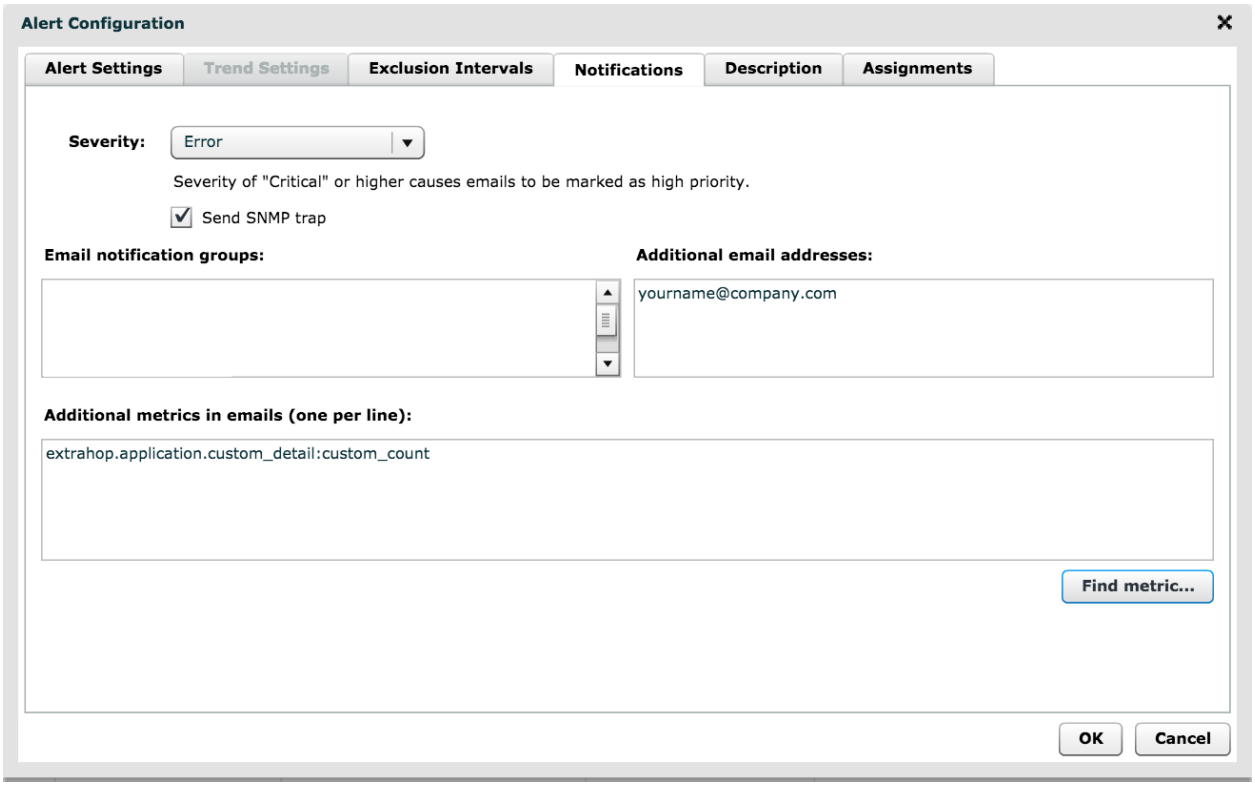
   Tip: The Email notification groups section displays all email groups configured in the Admin UI. You can select one or more groups that should receive alert notifications.

10. Complete the following steps to specify the metric to include in the email notification, which is the custom detail count metric named HTTP_error_allDetail.
    a) In the Additional metrics in emails section, click **Find metric**.
    b) Scroll down the page and select **extrahop.application.custom_detail:custom_count**.



    c) In the **Metric Name** field, type `HTTP_error_allDetail`, and then click **OK**.

The following figure displays the alert email notification settings we configured above:



11. Click **OK**.

Assign the alert configuration to a source

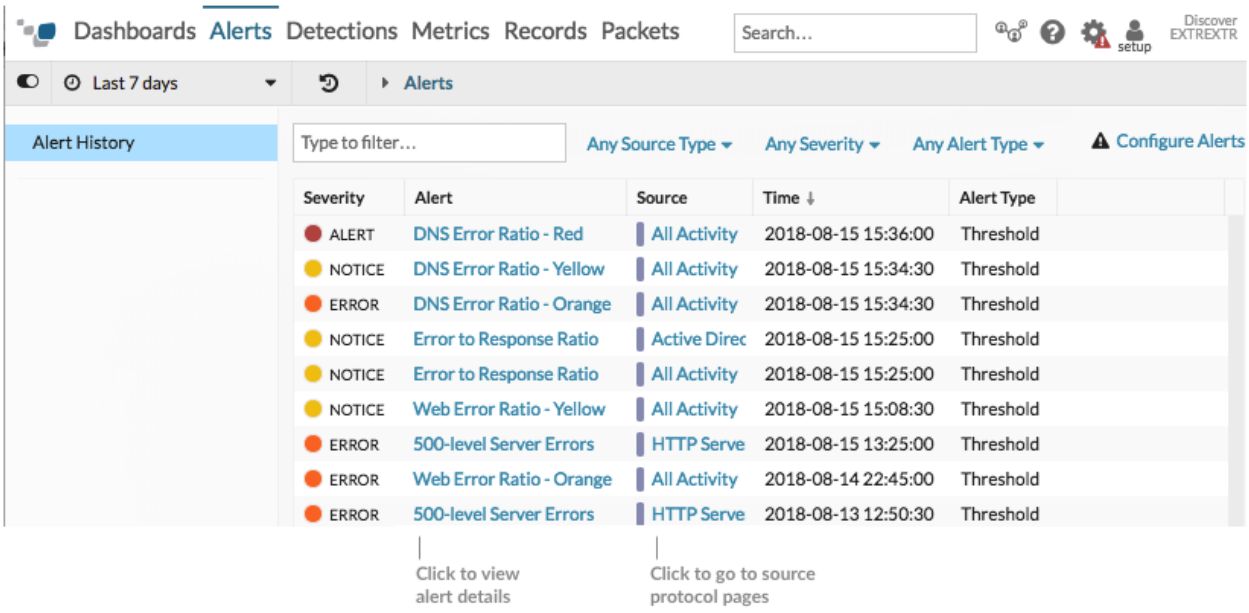
Similar to triggers, the system does not generate alerts until the alert configuration is assigned to at least one metric source. In this step, we will assign the alert configuration to the application named HTTP Server Errors that we created with the trigger script. The custom metrics we want the alert to track are committed to this application.

1. Click **Metrics** from the top menu.
2. Click **Applications**, and then select the **HTTP Server Errors** checkbox.
3. Click **Assign Alert** from the top of the page to open a list of alert configurations that are eligible for assignment.
4. Select the **500-level Server Errors** alert, and then click **Assign Alerts**.

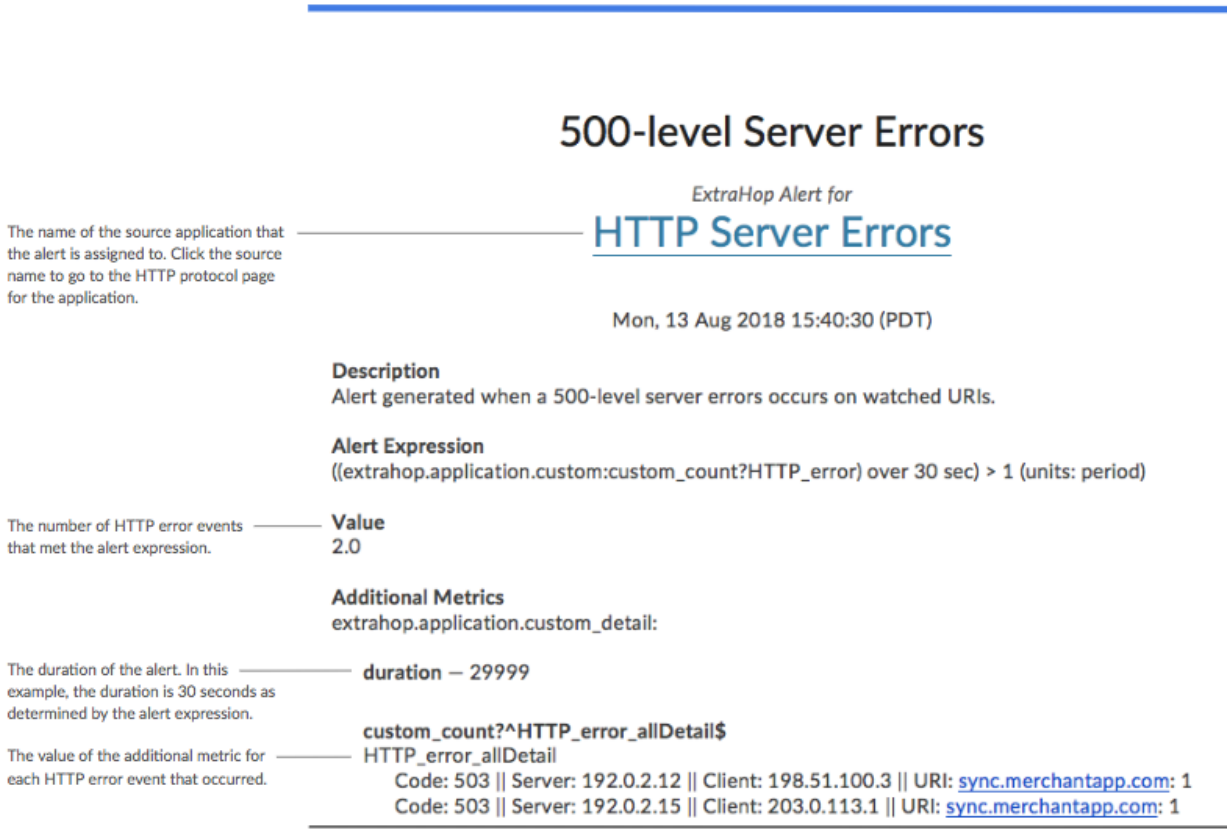
Check Alert History and view email notifications

Now that we've configured the alert and assigned it to a source, we can check if the alert has issued any entries.

Click **Alerts** from the top menu to view the Alert History and check for any alerts that were issued during the selected time interval, similar to the following figure:



When an alert is issued, a notification is sent to the specified email recipients The following email example shows that two HTTP error events occurred that met the conditions we set in the alert expression and provides additional information that helps us investigate the source of the errors:



In our example, we see that there were two 503 errors returned for the same URI across two different server IP addresses. A 503 status code might indicate an overloaded server that requires more CPU or memory resources to handle requests. By knowing the affected IP addresses you can immediately investigate potential problems on the listed servers.

Next steps

- Create charts to monitor your custom metrics on a dashboard or protocol page.
- Configure a trend alert to only issue alerts when server errors trend instead of for each error occurrence.
- Add an exclusion interval to your alert to suppress alerts during times when errors are expected.