Build a trigger

Published: 2020-02-22

Triggers provide expanded functionality of your ExtraHop system. With triggers, you can create custom metrics, generate and store records, or send data to a third-party system. Because you write the trigger script, you control the actions taken by the trigger upon specified system events.

To build a trigger, you must create a trigger configuration, write the trigger script, and then assign the trigger to one or more metric sources. The trigger will not run until all actions are completed.

Before you begin
Log in to the Discover or Command appliance with a user account that has the full write privileges required to create triggers.

If you are new to triggers, familiarize yourself with the trigger planning process, which will help you narrow the focus of your trigger, or determine whether you need a build a trigger at all. Then, run through the process of building a trigger by completing the Triggers Walkthrough.

Configure trigger settings

The first step to building a trigger is to provide a trigger name, determine whether debugging is enabled, and most importantly, identify which system events the trigger will run on.

1. Log into the Web UI on the ExtraHop Discover or Command appliance.
2. Click the System Settings icon and then click **Triggers**.
3. Click **New**, and then click the Configuration tab.
4. Specify the following trigger configuration settings:

Name

A name for the trigger.

Author

The name of the user that wrote the trigger. Default triggers display ExtraHop.

Description

An optional description of the trigger.

Status

A checkbox that enables or disables the trigger.

Debug

A checkbox that enables or disables debugging. If you add debug statements to the trigger script, this option enables you to view debug output in the runtime log when the trigger is running.

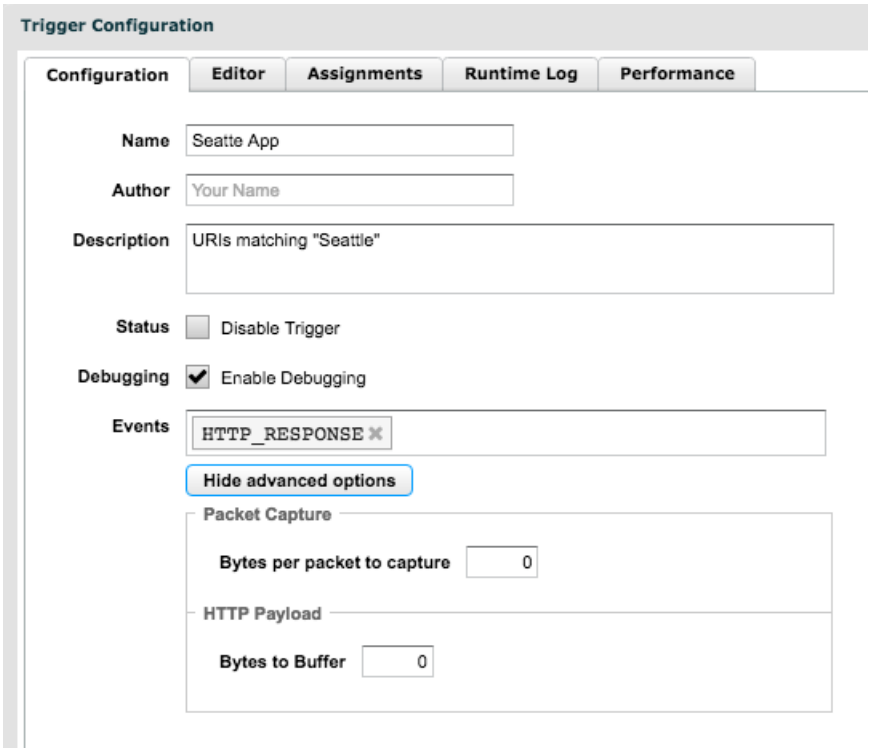
Events

The events on which the trigger runs. The trigger runs whenever one of the specified events occurs on an assigned device; therefore, you must assign at least one event to your trigger. You can click in the field or begin typing an event name to display a filtered list of available events.

Select advanced options

Advanced trigger options vary by the selected events. For example, if you select the `HTTP_RESPONSE` event, you can set the number of payload bytes to buffer on those events.

The following figure shows a sample configuration for a trigger than runs on HTTP responses:



Write a trigger script

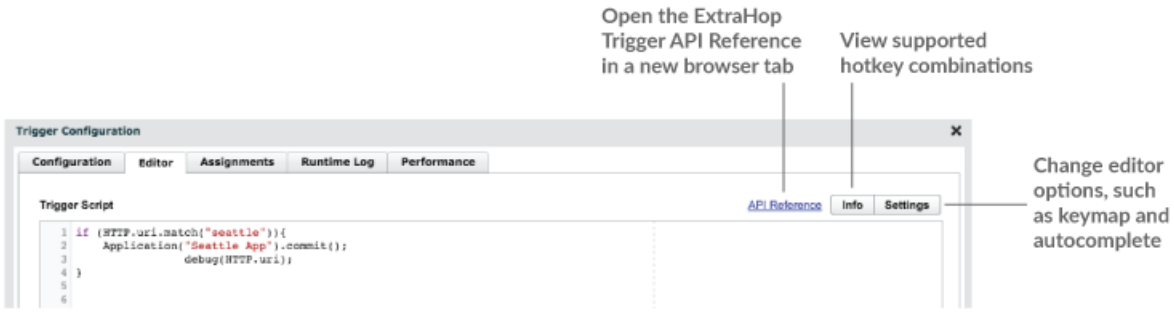
The trigger script specifies the instructions the trigger will carry out when a system event configured for the trigger occurs.

Before you begin

We recommend that you open the ExtraHop Trigger API Reference, which contains the events, methods, and properties you need for your trigger. A link is also available from the trigger editor window in the ExtraHop Web UI.

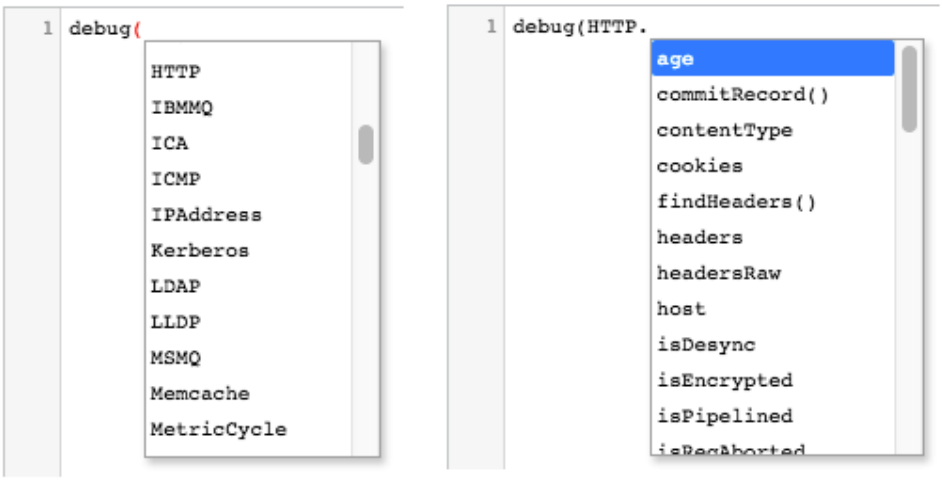
1. Log into the Web UI on the ExtraHop Discover or Command appliance.
2. Click the System Settings icon, and then click **Triggers**.
3. From the Trigger Configuration window, click the Editor tab.
4. Type the trigger script in JavaScript-like syntax with events, methods, and properties from the ExtraHop Trigger API Reference.

   The following figure shows a sample script entered on the Editor tab:



The editor provides an autocomplete feature that displays a list of properties and methods based on the selected class object. For example, press CTRL+Space in the editor to display a list of class objects, and

after you select a class, type a dot (.) to display a list of available properties and methods as shown in the following figure:

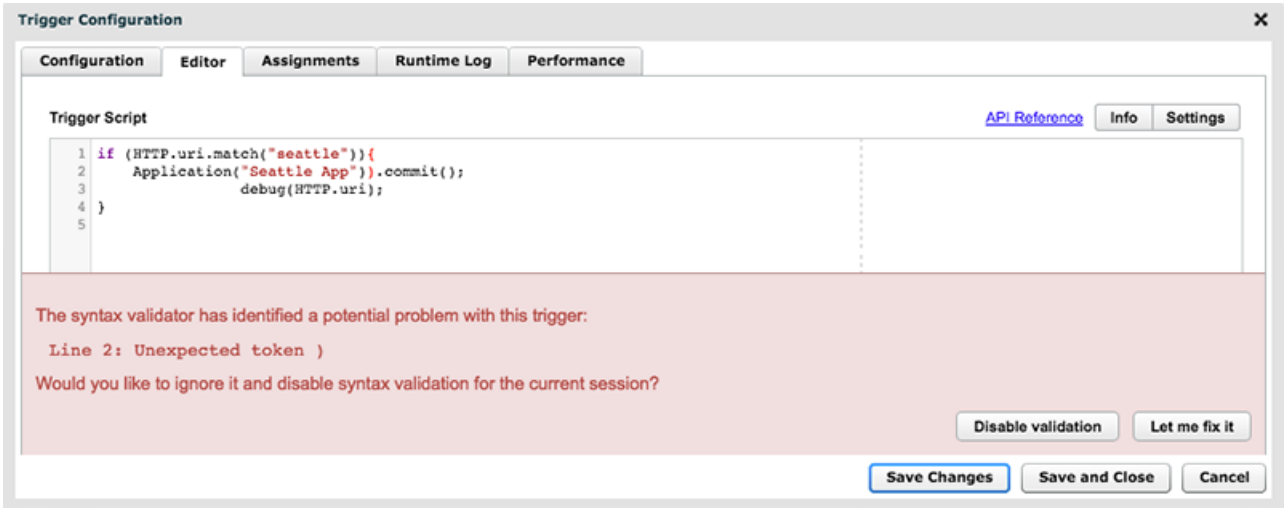


5. Click **Save Changes**.

The editor provides syntax validation of your script. When you save the trigger, the validator calls out any invalid actions, syntax errors, or deprecated elements in the script. If available, the validator displays replacements for deprecated elements. You cannot save the trigger until you fix your code or you disable syntax validation.

Warning: To avoid poor trigger performance, incorrect results, or a trigger that does not function, we strongly recommended that you fix the code or replace the deprecated element rather than disabling validation. Disabling validation applies only to the trigger you are editing; there is no option to disable validation globally.

The following figure shows a sample error message generated by the syntax validator:



After a new trigger is saved, the Runtime Log and Performance tabs are displayed.

Assign a trigger to a device

You can assign a trigger to one or more devices or to a device group. A trigger does not run until it is assigned to a device, and the trigger gathers metric data only from the devices to which it is assigned.

Warning: Running triggers on unnecessary devices and networks exhausts system resources. Minimize performance impact by assigning a trigger only to the specific sources that you need to collect data from.

Important: Triggers with the following events run whenever the event occurs. Triggers that only run on these events cannot be assigned to devices or device groups.

- ALERT_RECORD_COMMIT
- DETECTION_UPDATE
- METRIC_CYCLE_BEGIN
- METRIC_CYCLE_END
- METRIC_RECORD_COMMIT
- NEW_APPLICATION
- NEW_DEVICE
- SESSION_EXPIRE
- TIMER_30SEC

1. Log into the Web UI on the ExtraHop Discover or Command appliance.
2. Click **Metrics** from the top menu.
3. Click **Devices** or **Device Groups** in the left pane.
4. Select the checkbox for each device or device group you want to assign the trigger to.
5. Click the **Assign Trigger** icon from the top of the page.
6. Select the checkbox for each trigger you want to assign to the selected devices or device groups.
7. Click **Assign Triggers**.

The trigger runs on the selected devices whenever the trigger events occur.

Tip: You can also manage trigger assignments for a device from the device overview page. From the Manage Device section, click **Assignments** to add or remove trigger assignments from the device and to view which triggers are already assigned to the device.

Advanced trigger options

You must configure triggers to run on at least one event. Depending on the selected event, the Trigger Configuration window displays advanced configuration options. For example, selecting the `HTTP_RESPONSE` event enables you to set the number of payload bytes to buffer each time that event occurs on the system.

The following table describes available advanced options and the events that support each option.

Option	Description	Supported events
Bytes per packet to capture	Specifies the number of bytes to capture per packet. The capture starts with the first byte in the packet. Specify this option only if the trigger script performs packet capture. A value of 0 specifies that the capture should collect all bytes in each packet.	All events are supported except the following list: • ALERT_RECORD_COMMIT • METRIC_CYCLE_BEGIN • METRIC_CYCLE_END • FLOW_REPORT • NEW_APPLICATION • NEW_DEVICE • SESSION_EXPIRE
Bytes to Buffer	Specifies the minimum number of payload bytes to buffer.	• CIFS_REQUEST • CIFS_RESPONSE • HTTP_REQUEST • HTTP_RESPONSE

Option	Description	Supported events
		• ICA_TICK
Clipboard Bytes to Buffer	Specifies the number of bytes to buffer on a Citrix clipboard transfer.	• ICA_TICK
Metric Cycle	Specifies the length of the metric cycle, expressed in seconds. The following values are valid: • 30sec • 5min • 1hr • 24hr	• METRIC_CYCLE_BEGIN • METRIC_CYCLE_END • METRIC_RECORD_COMMIT
Metric Types	Specifies the metric type by the raw metric name, such as `extrahop.device.http_server`. Specify multiple metric types in a comma-delimited list.	• ALERT_RECORD_COMMIT • METRIC_RECORD_COMMIT
Per Turn	Enables packet capture on each flow turn. Per-turn analysis continuously analyzes communication between two endpoints to extract a single payload data point from the flow. If this option is enabled, any values specified for the **Client matching string** and **Server matching string** options are ignored.	• SSL_PAYLOAD • TCP_PAYLOAD
Client port min	Specifies the minimum port number of the client port range. Valid values are 0 to 65535. A value of 0 specifies matching of any port.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
Client port max	Specifies the maximum port number of the client port range. Valid values are 0 to 65535. Any value specified for this option is ignored if the value of the **Client port min** option is 0.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
Client bytes to buffer	Specifies the number of client bytes to buffer. The value of this option cannot be set to 0 if the value of the **Server bytes to buffer** option is also set to 0.	• SSL_PAYLOAD • TCP_PAYLOAD

Option	Description	Supported events
Client matching string	Specifies the format string that indicates when to begin buffering client data. Any value specified for this option is ignored if the **Per Turn** option is enabled.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
Server port min	Specifies the minimum port number of the server port range. Valid values are 0 to 65535. A value of 0 specifies matching of any port.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
Server port max	Specifies the maximum port number of the server port range. Valid values are 0 to 65535. Any value specified for this option is ignored if the value of the **Server port min** option is 0.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
Server bytes buffer	Specifies the number of server bytes to buffer. The value of this option cannot be set to 0 if the value of the **Client bytes to buffer** option is also set to 0.	• SSL_PAYLOAD • TCP_PAYLOAD
Server matching string	Specifies the format string that indicates when to begin buffering data. Returns the entire packet upon a string match. Any value specified for this option is ignored if the **Per Turn** option is enabled.	• SSL_PAYLOAD • TCP_PAYLOAD • UDP_PAYLOAD
All UDP Datagrams	Enables capture of all UDP datagrams.	• UDP_PAYLOAD
Run FLOW_CLASSIFY on expired flows	Enables running the event upon expiration to accumulate metrics for flows that were not classified before expiring.	• FLOW_CLASSIFY