ExtraHop
Leitfaden mit bewährten Methoden für Trigger

Veröffentlicht: 2023-09-14

Schreiben eines Auslöser Das Sammeln von Metriken ist eine leistungsstarke Methode zur Überwachung Ihrer Anwendungs- und Netzwerkleistung. Trigger verbrauchen jedoch Systemressourcen und können die Systemleistung beeinträchtigen — ein schlecht geschriebener Auslöser kann zu unnötiger Systemlast führen. Bevor Sie einen Auslöser schreiben, sollten Sie auswerten, was Ihr Auslöser bewirken soll, ermitteln, welche Ereignisse und Geräte zum Extrahieren der benötigten Daten erforderlich sind, und feststellen, ob es bereits eine Lösung gibt.

Identifizieren Sie, was Sie wissen möchten

Identifizieren Sie eindeutig, welche Informationen Sie wissen möchten oder was der Auslöser bewirken soll, z. B. in den folgenden Beispielen:

- Wann laufen meine SSL-Zertifikate ab?
- Erhält mein Netzwerk Verbindungen über nicht autorisierte Ports?
- Wie viele langsame Transaktionen verzeichnet mein Netzwerk?
- Welche Daten möchte ich über einen offenen Datenstrom an Splunk senden?

Indem Sie das Problem identifizieren, das Sie lösen möchten, können Sie einen Auslöser besser darauf ausrichten, nur die Informationen zu sammeln, die Sie benötigen, und so unnötige Leistungseinbußen vermeiden.

Konzentrieren Sie den Auslöser auf eine logische Funktion

Ein Auslöser sollte eine logische Funktion ausführen. Wenn Sie einen Auslöser benötigen, der eine andere Aufgabe ausführt, erstellen Sie einen zweiten Auslöser. Ein Auslöser, der mehrere Aufgaben ausführt, die nichts miteinander zu tun haben, verbraucht mehr Ressourcen.

Suchen Sie im Metrikkatalog nach Metriken

Lesen Sie unbedingt die integrierten Metriken in der Metrischer Katalog. Integrierte Messwerte belasten das System nicht zusätzlich und enthalten möglicherweise bereits die Informationen, die Sie benötigen.

Identifizieren Sie, welche Systemereignisse die Daten erzeugen, die Sie sammeln möchten

Beispielsweise kann ein Auslöser, der die Aktivität von Cloud-Anwendungen in Ihrer Umgebung überwacht, bei HTTP-Antworten und beim Öffnen und Schließen von SSL-Verbindungen ausgeführt werden.

Identifizieren Sie die Geräte oder Netzwerke, die Sie überwachen und von denen Sie Messwerte sammeln möchten

Ein Auslöser verbraucht weniger Systemressourcen, wenn Sie auf bestimmte Geräte und nicht auf alle Geräte eines bestimmten Typs oder einer bestimmten Gruppe abzielen. Beispielsweise sollte ein Auslöser, der nach langsamen Antworten aus Ihrem Online-Katalog sucht, nur HTTP-Servern zugewiesen werden, die Katalogtransaktionen abwickeln, und nicht allen HTTP-Servern.

Bestimmen Sie, wie Sie die vom Auslöser gesammelten Daten visualisieren oder speichern möchten

Sie können beispielsweise Metriken in einem Dashboard anzeigen, Datensätze an einen Recordstore senden oder Daten an ein Drittanbietersystem senden.

Ermitteln Sie, ob bereits ein Auslöser vorhanden ist

Es ist möglich, dass bereits ein Auslöser vorhanden ist, der Ihren Anforderungen entspricht oder leicht geändert werden kann. Beginnen Sie immer mit einem bereits vorhandenen Auslöser, wann immer dies möglich ist. Durchsuche die ExtraHop Community-Foren für verfügbare Trigger.

Optimieren Sie die Trigger-Konfiguration

Die Konfigurationsoptionen, die beim Erstellen oder Ändern eines Auslöser verfügbar sind, können sich auf die Leistung und Effizienz des Auslöser auswirken. ExtraHop empfiehlt die folgenden Methoden zur Optimierung und Verbesserung der Triggerleistung.

Nutzen Sie die erweiterten Trigger-Optionen

Wenn erweiterte Trigger-Optionen sind für das Ereignis verfügbar, für das der Auslöser ausgeführt wird. Wir empfehlen Ihnen, entsprechende Optionen zu konfigurieren, um den Fokus des Auslöser einzugrenzen und Leistung und Ergebnisse zu verbessern. Wenn Sie beispielsweise einen Auslöser erstellen , der auf dem `SSL_PAYLOAD` Bei diesem Ereignis können Sie minimale und maximale Portnummern angeben, sodass der Auslöser nur Daten von Transaktionen innerhalb des angegebenen Portbereichs sammelt.

Weisen Sie den Auslöser minimalen Ressourcen zu

Verhindern Sie, dass der Auslöser unnötig ausgeführt wird, indem Sie den Auslöser so wenigen Quellen wie möglich zuweisen, z. B. Geräten. Wählen Sie nicht **Allen Geräten zuweisen** bei der Zuweisung des Auslöser und vermeiden Sie Zuweisungen zu großen Gerätegruppen.

Debuggen Sie nur beim Testen

Das Debuggen sollte nur aktiviert werden, während Sie aktiv an Ihrem Auslöser arbeiten. Debugging ist nützlich, um zu testen, ob der Trigger ausgeführt wird und die erwarteten Daten sammelt. Debuggen beansprucht jedoch unnötig Ressourcen und sollte in einer Produktionsumgebung vermieden werden.

Schreiben Sie ein optimiertes Trigger-Skript

In den folgenden Abschnitten werden allgemeine bewährte Methoden für die Codierung sowie Richtlinien für die Arbeit mit API-Komponenten und -Objekten wie Sitzungstabellen und Datensätzen behandelt.

Bewährte Programmierpraktiken anwenden

Beim Schreiben eines Trigger-Skripts empfehlen wir die folgenden bewährten Methoden zur Codierung, um die Trigger-Leistung zu optimieren und zu verbessern.

Auslöserausnahmen und Fehler sofort beheben

Ausnahmen beeinträchtigen die Leistung erheblich. Wenn Ausnahmen oder Fehler im Debug-Log eines Triggers auftauchen, beheben Sie diese Probleme sofort. Wenn es Ausnahmen gibt, die nicht einfach vermieden werden können, können Sie versuchen, die Ausnahme mit einem zu behandeln `try/catch` Aussage. Die `try/catch` Die Anweisung sollte in eine kleine Funktion eingebunden werden, die keine andere Operation ausführt.

Im Beispiel unten ist der `JSON.parse` Die Funktion führt zu einer Ausnahme, wenn die Eingabe nicht in wohlgeformter JSON-Syntax erfolgt. Da es keine Möglichkeit gibt, vor dem Parsen zu überprüfen, ob die JSON-Syntax wohlgeformt ist, ist diese Aufgabe in eine kleine Hilfsfunktion eingebunden.

```
function parseJSON(jsonString) {
    try {
        return JSON.parse(jsonString);
    } catch (e) {
        return null;
    }
}
let obj = parseJSON(HTTP.payload);
```

Bewahren Sie Aktionen dort auf, wo sie im Skript benötigt werden

Verschieben Sie Aktionen, wie Zeichenkettenoperationen und Zugriff auf Eigenschaften, in den exklusivsten Zweig, für den die Aktion erforderlich ist.

```
// inefficient: The cookies variable may not be needed
let cookies = HTTP.cookies;
if (HTTP.method === 'POST') {
 // process cookies
}

// optimized: Cookies are not accessed unless they are needed
if (HTTP.method === 'POST') {
 let cookies = HTTP.cookies;
 // process cookies
}
```

Lokales Speichern von Objekten, auf die häufig zugegriffen wird

Sparen Sie CPU-Zyklen, indem Sie Anwendungen, Standorte, Geräte und andere Objekte, auf die mehrfach zugegriffen wird, lokal speichern.

```
// store object locally
let app = Application('example');
app.commit();
app.metricAddCount('custom', 1);
```

Vermeiden Sie Variablen in der Debug-Anweisung

Nehmen Sie eine Variable nicht ausschließlich zum Debuggen in die Debug-Anweisung auf, wenn Sie auf eine Nutzlast oder ein anderes großes Objekt zugreifen. Wenn Sie Variablen aus der Debug-Anweisung ausschließen, können Sie die Debug-Zeile auskommentieren und den Zugriff auf die Payload-Informationen aufrechterhalten.

Nehmen Sie die Funktion exit () nicht in das Skript auf

Fügen Sie ein `return` Aussage statt `exit()` globale Funktion, die aufgrund schlechter Leistung veraltet ist.

```
// avoid: slows the trigger down
HTTP.uri || exit();

// recommended
if (!HTTP.uri) return;
```

Hinweis: Das Verhalten der `return` Die Anweisung ist anders, wenn sie sich innerhalb einer verschachtelten Funktion befindet; in einer verschachtelten Funktion ist die `return` Die Anweisung beendet nur die aktuelle Funktion, nicht den gesamten Auslöser.

Nehmen Sie die Funktion eval () nicht in das Skript auf

Trigger sollten nicht versuchen, Code zur Laufzeit dynamisch zu generieren; daher `eval()` Diese Funktion wird von der ExtraHop Trigger API nicht unterstützt. Die Aufnahme der Funktion in den Auslöser kann zu unvorhersehbaren Laufzeitfehlern führen.

Reguläre Ausdrücke priorisieren

Wenden Sie das an `test()` Methode mit regulären Ausdrücken anstelle der `match()` Methode, wenn möglich. Das Extrahieren eines Werts mit einem regulären Ausdruck ist oft schneller als das vollständige Parsen von XML- oder Abfragezeichenfolgen.

```
// slower search
let user = HTTP.parseQuery(payload).user;

// faster search
```

```
let match = /user=(.*?)\&/i.exec(payload);
let user = match ? match[1] : null;
```

Wenden Sie strenge Gleichheitsoperatoren an

Verwenden Sie für Vergleiche strenge Gleichheitsoperatoren wie Triple Equals (===) statt loser Gleichheit (==), wann immer möglich. Strikte Gleichheitsoperatoren sind schneller, da sie nicht versuchen, Typzwänge zu erzwingen, z. B. IP-Adressen vor dem Vergleich in Zeichenketten umzuwandeln.

Schließen Sie keine nicht deklarierten Variablen ein

Deklarieren Sie Variablen mit `let`, `var`, oder `const` Anweisungen; schließen Sie niemals nicht deklarierte Variablen ein.

```
// inefficient: The variable "cookies" is never declared
cookies = HTTP.cookies;

// optimized: The variable "cookies" is declared
let cookies = HTTP.cookies;
```

Organisiere das Skript mit Funktionen

Funktionen sind gut für die Leistung und die Codeorganisation. Definieren Sie Funktionen am Anfang des Auslöser, wenn Ihr Auslöser dieselbe Operation an verschiedenen Stellen im Trigger-Skript ausführt oder wenn Triggeroperationen logisch in einzelne Teile aufgeteilt werden können.

Organisieren Sie Hilfsfunktionen oben im Skript

Definieren Sie Hilfsfunktionen oben im Triggerskript statt unten, um die Lesbarkeit zu verbessern und dem ExtraHop-System zu helfen, Ihren Code besser zu verstehen und auszuführen.

Hinzufügen von Sitzungstabellen

Die Sitzungstabelle verwendet weltweit einfache Datentypen (wie Zeichenketten) für alle Trigger und Flows. Beachten Sie beim Hinzufügen von Sitzungstabellen zu Ihrem Triggerskript die folgenden bewährten Methoden.

Erhöhen Sie die Anzahl in Sitzungstabellen

Rufen Sie beim Zählen den `Session.increment` Funktion statt `Session.lookup` oder `Session.replace` Funktionen. Die `Session.increment` Die Funktion ist atomar und daher garantiert, dass sie über mehrere Trigger-Threads hinweg korrekt ist.

Erfassen Sie keine eindeutigen Zählmetriken mit einer Sitzungstabelle

Eine Sitzungstabelle ist kein optimales Tool zum Zählen der Anzahl eindeutiger Werte wie IP-Adressen, Ports oder Benutzernamen und kann zu Leistungsproblemen führen. Übergeben Sie stattdessen einen benutzerdefinierten eindeutige Anzahl Metrik mit einer der folgenden Methoden:

```
metricAddDistinct (
   name: String,
   item: String|Number|IPAddress
)

metricAddDetailDistinct (
   name: String,
    key: String|IPAddress,
   item: String|Number|IPAddress
)
```

Erstellen oder iterieren Sie keine sehr großen Arrays oder Objekte

Das Hinzufügen großer Arrays oder Objekte zu einer Sitzungstabelle kann die Leistung beeinträchtigen. Fügen Sie beispielsweise nicht mehrere ablaufende Schlüssel hinzu, die eine `notify=true` Wert, in einem Auslöser, der auf dem läuft `SESSION_EXPIRE` Ereignis. Rufen Sie

außerdem nicht an `JSON.parse` oder `JSON.stringify` Methoden für große Objekte in der Sitzungstabelle.

Werte im Flow Store platzieren

Der Flow-Speicher sammelt ereignisübergreifende Werte in einem einzelnen Flow, z. B. einem Anforderungs-/Antwortpaar. Beachten Sie die folgenden bewährten Methoden, wenn Sie Werte in einem Trigger-Skript im Flow-Speicher platzieren.

Ermitteln Sie, ob bereits Werte verfügbar sind

Vor der Arbeit mit dem `Flow.store` Eigenschaft, überprüfen Sie, ob die gewünschten Werte bereits direkt in der ExtraHop Trigger API verfügbar sind. Neuere Firmware-Versionen haben die Notwendigkeit eines Flow-Speichers für eine Reihe von Anforderungseigenschaften, die häufig in der Antwort verwendet wurden, überflüssig gemacht, wie z. `HTTP.query` und `DB.statement`.

Nur benötigte Werte zum Flow Store hinzufügen

Vermeiden Sie das Hinzufügen großer Objekte zum `Flow.store` Eigenschaft, wenn Sie nur eine Teilmenge der Objektwerte benötigen. Der folgende Code speichert beispielsweise nur die `TTL` Immobilienwert aus dem `DNS.answers` Objekt, anstatt alle Eigenschaftswerte aus dem zu speichern `DNS.answers` Objekt:

```
Flow.store.ttl = DNS.answers[0].ttl;
```

Werte löschen, wenn sie nicht mehr benötigt werden

Klar `Flow.store` Eigenschaftswerte, wenn sie nicht mehr benötigt werden, indem der Eigenschaftswert auf gesetzt wird `null`. Durch das Löschen des Flow-Speichers können Fehler vermieden werden, bei denen der Auslöser denselben Wert mehrmals verarbeitet.

Flow-Speicherwerte nicht zwischen Triggern teilen

Greifen Sie nicht auf `Flow.store` Objekt zum Teilen von Werten zwischen verschiedenen Triggern, die auf demselben Ereignis ausgeführt werden. Die Reihenfolge, in der Trigger ausgeführt werden, wenn ein Ereignis eintritt, ist nicht garantiert. Wenn ein Auslöser vom Wert des abhängt `Flow.store` Bei einer Eigenschaft eines zweiten Auslöser entspricht der Wert möglicherweise nicht den Erwartungen und kann im ersten Trigger zu falschen Ergebnissen führen.

Erstellen von benutzerdefinierten Datensätzen

Benutzerdefinierte Metriken bieten die Flexibilität, die benötigten Daten zu erfassen, sofern die Daten nicht bereits von den integrierten Protokollmetriken auf dem ExtraHop-System bereitgestellt werden. Beachten Sie die folgenden bewährten Methoden, wenn Sie benutzerdefinierte Metriken über einen Auslöser erstellen.

Generieren Sie keine benutzerdefinierten Metriknamen dynamisch

Das dynamische Generieren von benutzerdefinierten Metriknamen beeinträchtigt die Leistung und verhindert möglicherweise, dass Metriken automatisch im Metrikkatalog erkannt werden.

Konvertieren Sie IP-Adressen nicht in eine Zeichenfolge, bevor Sie sie zu benutzerdefinierten Metriken hinzufügen

Wenn Sie eine IP-Adresse in eine Zeichenfolge konvertieren, bevor Sie sie als Detail in einer benutzerdefinierten Metrik hinzufügen, kann das ExtraHop-System keine mit der IP-Adresse verknüpften Attribute abrufen, z. B. das zugehörige Gerät oder den Hostnamen.

```
// avoid
Application("SampleApp").metricAddDetailCount("reqs.byClient",
 Flow.client.ipaddr.toString(), 1);

// recommended
Application("SampleApp").metricAddDetailCount("reqs.byClient",
 Flow.client.ipaddr, 1);
```

Daten in Datensätzen speichern

Mit Datensätzen können Sie strukturierte Informationen über Transaktions-, Nachrichten- und Netzwerkflüsse speichern und abrufen. Beachten Sie beim Erstellen und Speichern von Datensätzen über einen Auslöser die folgenden bewährten Methoden.

Beschränken Sie den Zugriff auf Eigenschaften über das Datensatzobjekt

Vermeiden Sie den Zugriff auf Eigenschaften über ein Datensatzobjekt, wie `HTTP.record`, wenn die Eigenschaften im Protokollobjekt oder Flow-Objekt verfügbar sind. Wenn Sie auf das Datensatzobjekt zugreifen, werden die Datensatzdaten im Speicher zugewiesen. Greifen Sie stattdessen auf das Datensatzobjekt zu, wenn Sie Daten in einem geänderten benutzerdefinierten Datensatz speichern.

```
// avoid
let uri = HTTP.record.uri;

// recommended
let uri = HTTP.uri;
```

Konvertieren Sie IP-Adressen nicht in Zeichenketten, bevor Sie sie in Datensätzen speichern

Das ExtraHop-System filtert und sortiert nach IP-Adressen, was nicht auf Zeichenketten wie CIDR-Blöcke für IPv4-Adressen angewendet werden kann.

Zugreifen auf den Datenspeicher

Mit Datenspeicher-Triggern können Sie auf aggregierte Messwerte zugreifen, die für transaktionsbasierte Ereignisse nicht verfügbar sind, z. B. `HTTP_REQUEST`. Beachten Sie die folgenden bewährten Methoden, wenn Sie über einen Auslöser auf den Datenspeicher zugreifen.

Datenspeicher-Trigger sparsam erstellen

Die Ressourcen für Datenspeicher-Trigger sind sehr begrenzt. Vermeiden Sie nach Möglichkeit das Ausführen von Triggern für die folgenden Datenspeicherereignisse:

- ALERT_RECORD_COMMIT
- METRIC_RECORD_COMMIT
- METRIC_CYCLE_BEGIN
- ENDE DES METRISCHEN ZYKLUS

Wenn Sie beispielsweise nur in regelmäßigen Abständen auf eine Sitzungstabelle zugreifen möchten, führen Sie den Auslöser auf `SESSION_EXPIRE` Ereignis statt `METRIC_CYCLE_END` Ereignis.

Konfigurieren Sie erweiterte Optionen beim Festschreiben von Metrikdatensätzen

Stellen Sie sicher, dass erweiterte Trigger-Optionen sind für Trigger konfiguriert, die auf dem `METRIC_RECORD_COMMIT` Ereignis. Insbesondere empfehlen wir, einen 30-Sekunden-Metrikzyklus für Trigger zu konfigurieren, die auf dem `METRIC_RECORD_COMMIT` Ereignis.

Auslöserleistung auswerten

Sie können die Auswirkungen von Triggern, die auf Ihrem ExtraHop-System ausgeführt werden, auf verschiedene Arten überwachen und bewerten. Sie können Leistungsinformationen für einen einzelnen Auslöser und mehrere Diagramme anzeigen, die die kollektiven Auswirkungen all Ihrer Trigger auf das System angeben.

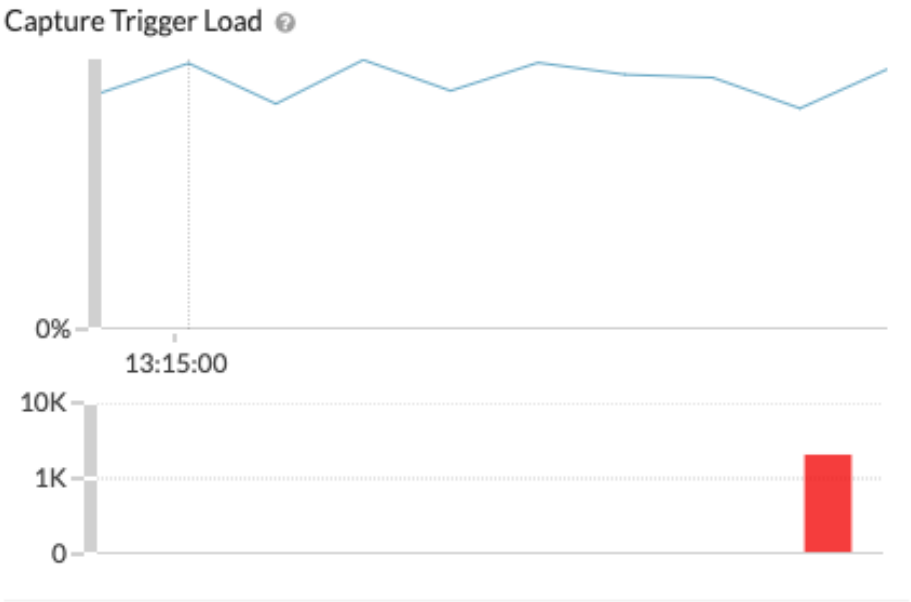
Die Leistung eines einzelnen Auslöser anzeigen

Auf der Registerkarte Debug-Protokoll im Bereich Auslöser bearbeiten können Sie überprüfen, ob ein Trigger Fehler oder Ausnahmen aufweist. Rufen Sie die Trigger-Seite auf, indem Sie auf das Symbol Systemeinstellungen klicken im ExtraHop-System.

PROBLEMS 0 0 DEBUG LOG

```
[Wed Jun 12 12:36:57]  Committing Database record for event type DB_RESPONSE.
[Wed Jun 12 12:37:27]  Committing Database record for event type DB_RESPONSE.
[Wed Jun 12 12:37:47]  Committing Database record for event type DB_RESPONSE.
[Wed Jun 12 12:38:25]  Committing Database record for event type DB_RESPONSE.
[Wed Jun 12 12:38:53]  Committing Database record for event type DB_RESPONSE.
[Wed Jun 12 12:39:28]  Committing Database record for event type DB_RESPONSE.
[Wed Jun 12 12:39:58]  Committing Database record for event type DB_RESPONSE.
[Wed Jun 12 12:40:26]  Committing Database record for event type DB_RESPONSE.
[Wed Jun 12 12:40:58]  Committing Database record for event type DB_RESPONSE.
```

Sie können die Leistungskosten eines laufenden Triggers im Diagramm „Capture Trigger Load“ auf der Registerkarte „Trigger bearbeiten“ einsehen. Das Diagramm zeigt ein Trigger-Leistungsdiagramm , das die Anzahl der Zyklen aufzeichnet, die der Auslöser innerhalb eines bestimmten Zeitintervalls verbraucht hat. Sie können den Mauszeiger über einen Datenpunkt bewegen, um wichtige Leistungskennzahlen zu einem einzigen Zeitpunkt anzuzeigen.



Der Hover-Tipp enthält die folgenden Informationen:

- Die meisten und die wenigsten Zyklen, die der Auslöser zur Verarbeitung eines einzelnen Ereignis verbraucht hat.
- Die Häufigkeit, mit der der Auslöser ausgeführt wurde, und wie oft der Auslöser ausgeführt wurde, im Vergleich zu allen Triggern, die im gleichen Zeitraum ausgeführt wurden.
- Die Gesamtzahl der vom Auslöser verbrauchten Zyklen und der Prozentsatz der verbrauchten Zyklen im Vergleich zu allen Triggern, die im gleichen Zeitraum ausgeführt wurden.

Die Leistung aller Trigger auf dem System anzeigen

Die Seite „Systemstatus“ enthält mehrere Diagramme, die einen Überblick über die Trigger bieten, die auf dem ExtraHop-System ausgeführt werden. Rufen Sie die Seite Systemstatus auf, indem Sie im ExtraHop-System auf das Symbol Systemeinstellungen klicken.

Anhand der in den folgenden Richtlinien beschriebenen Diagramme zur Systemintegrität können Sie nach Problemen suchen, die sich auf die Systemleistung auswirken oder zu falschen Daten führen können.

Stellen Sie sicher, dass der Auslöser ausgeführt wird

Die Einzelheiten zum Auslöser In einem Diagramm werden alle Trigger angezeigt, die auf dem System ausgeführt werden. Wenn der Auslöser, den Sie gerade erstellt oder geändert haben, nicht aufgeführt ist, liegt möglicherweise ein Problem mit dem Triggerskript vor.

Überwachen Sie auf unerwartete Aktivitäten

Die Trigger wird ausgeführt und gelöscht Ein Diagramm kann Ausbrüche von Triggeraktivitäten anzeigen, die auf ineffizientes Verhalten durch einen oder mehrere Auslöser hinweisen können. Wenn irgendwelche Bursts angezeigt werden, schauen Sie sich die Trigger wird von Trigger ausgeführt Diagramm, um jeden Auslöser zu finden, der mehr Ressourcen als der Durchschnitt verbraucht. Dies kann darauf hindeuten, dass der Auslöser über ein schlecht optimiertes Skript verfügt, das die Leistung beeinträchtigt.

Suchen Sie nach unbehandelten Trigger-Ausnahmen

Die Ausnahmen nach Trigger auslösen In einem Diagramm werden alle Ausnahmen angezeigt, die durch Trigger verursacht wurden. Ausnahmen tragen wesentlich zu Problemen mit der Systemleistung bei und sollten sofort behoben werden.

Suchen Sie nach Triggern, die aus der Warteschlange gelöscht wurden

Die Trigger wird ausgeführt und gelöscht In einem Diagramm wird die Anzahl der Trigger angezeigt, die aus der Trigger-Warteschlange gelöscht wurden. Eine häufige Ursache für gelöschte Trigger ist ein lang andauernder Auslöser, der den Ressourcenverbrauch dominiert. Ein gesundes System sollte zu jeder Zeit 0 Tropfen enthalten.

Überwachen Sie den Ressourcenverbrauch

Die Last auslösen Das Diagramm verfolgt die Nutzung aller verfügbaren Ressourcen durch Trigger. Eine hohe Belastung beträgt ungefähr 50%. Achten Sie auf Verbrauchsspitzen, die darauf hindeuten können, dass ein neuer Auslöser eingeführt wurde oder dass bei einem vorhandenen Auslöser Probleme auftreten.

Anhand der folgenden Diagramme können Sie überwachen, ob Ihre Datenspeicher-Trigger, auch Bridge-Trigger genannt, ordnungsgemäß ausgeführt werden:

- Der Datenspeicher-Trigger wird ausgeführt und gelöscht
- Laden des Datenspeicher-Triggers
- Datenspeicherauslöserausnahmen nach Trigger

Sehen Sie die Häufig gestellte Fragen zur Systemgesundheit um häufig gestellte Fragen dazu zu lesen, wie Systemstatusdiagramme Ihnen bei der Bewertung der Triggerleistung auf Ihrem ExtraHop-System helfen können.

Trigger-Ressourcen anzeigen

Weitere Informationen zu Triggern und der ExtraHop Trigger API finden Sie in den folgenden Ressourcen:

Dokumentation

- ExtraHop Trigger API-Referenz
- Erste Schritte mit Triggern Abschnitt der Leitfaden zur Web-Benutzeroberfläche.

Exemplarische Vorgehensweisen

- Exemplarische Vorgehensweise für Auslöser: Erstellen Sie einen Trigger zum Sammeln benutzerdefinierter Metriken für HTTP 404-Fehler
- UPA-Komplettlösung: Erstellen Sie einen Auslöser, um Antworten auf NTP-Monlist-Anfragen zu überwachen
- Exemplarische Vorgehensweise zum Öffnen von Datenströmen: Konfiguration eines offenen Datenstroms zum Senden von Metrikdaten an AWS Cloudwatch

Schulung

- Einen Trigger planen
- Einen einfachen Trigger erstellen
- App-Container erstellen und verwenden
- Einen Multi-Event-Trigger erstellen

Gemeinschaft

- Die ExtraHop Community-Foren