ExtraHop
Guide des meilleures pratiques pour les déclencheurs

Publié: 2023-09-19

L'écriture d'un déclencheur pour collecter des métriques est un moyen puissant de surveiller les performances de votre application et de votre réseau. Cependant, les déclencheurs consomment des ressources système et peuvent affecter les performances du système - un déclencheur mal écrit peut entraîner une charge système inutile. Avant d'écrire un déclencheur, vous devez évaluer ce que vous voulez qu'il accomplisse, identifier les événements et les périphériques nécessaires pour extraire les données dont vous avez besoin et déterminer si une solution existe déjà.

Identifier ce que vous voulez savoir

Identifiez clairement les informations que vous souhaitez connaître ou ce que vous voulez que le déclencheur accomplisse, comme dans les exemples suivants :

- Quand mes certificats SSL expireront-ils ?
- Mon réseau reçoit-il des connexions sur des ports non autorisés ?
- Combien de transactions lentes mon réseau connaît-il ?
- Quelles sont les données que je souhaite envoyer à Splunk par le biais d'un flux de données ouvert ?

En identifiant le problème que vous essayez de résoudre, vous pouvez mieux cibler un déclencheur pour collecter uniquement les informations dont vous avez besoin et éviter de subir un impact inutile sur les performances.

Concentrer le déclencheur sur une fonction logique

Un déclencheur doit remplir une seule fonction logique. Si vous avez besoin d'un déclencheur qui effectue une tâche différente, créez-en un second. Un déclencheur qui effectue plusieurs tâches sans lien entre elles consomme davantage de ressources.

Recherche de métriques dans le catalogue des métriques

Veillez à consulter les métriques intégrées dans le catalogue de métriques. Les métriques intégrées ne créent pas de charge supplémentaire sur le système et peuvent déjà contenir les informations dont vous avez besoin.

Identifiez les événements système qui produisent les données que vous souhaitez collecter

Par exemple, un déclencheur qui surveille l'activité des applications en nuage dans votre environnement peut être exécuté sur les réponses HTTP et sur l'ouverture et la fermeture des connexions SSL.

Identifiez les dispositifs ou les réseaux que vous souhaitez surveiller et dont vous souhaitez collecter les métriques

Un déclencheur consomme moins de ressources système si vous ciblez des périphériques spécifiques plutôt que tous les périphériques d'un type ou d'un groupe particulier. Par exemple, un déclencheur qui recherche les réponses lentes de votre catalogue en ligne ne doit être affecté qu'aux serveurs HTTP qui traitent les transactions du catalogue et non à tous les serveurs HTTP.

Déterminez comment vous souhaitez visualiser ou stocker les données collectées par le déclencheur

Par exemple, vous pouvez afficher les mesures sur un tableau de bord, envoyer des enregistrements à un magasin d'enregistrements ou envoyer des données à un système tiers.

Déterminer si un déclencheur existe déjà

Il est possible qu'il existe déjà un déclencheur qui réponde à vos besoins ou qui puisse être facilement modifié ; commencez toujours par un déclencheur préexistant dans la mesure du possible. Recherchez les déclencheurs disponibles dans les forums de la communauté ExtraHop.

Optimiser la configuration du déclencheur

Les options de configuration disponibles lorsque vous créez ou modifiez un déclencheur peuvent affecter ses performances et son efficacité. ExtraHop recommande les pratiques suivantes pour optimiser et améliorer les performances des déclencheurs.

Tirer parti des options avancées du déclencheur

Si des options de déclenchement avancées sont disponibles pour l'événement sur lequel le déclencheur s'exécute, nous vous recommandons de configurer les options applicables afin de restreindre l'objectif du déclencheur et d'améliorer les performances et les résultats. Par exemple, si vous créez un déclencheur qui s'exécute sur l'événement SSL_PAYLOAD, vous pouvez spécifier des numéros de port minimum et maximum afin que le déclencheur ne collecte que les données des transactions comprises dans la plage de ports spécifiée.

Affecter le déclencheur à des ressources minimales

Pour éviter que le déclencheur ne s'exécute inutilement, affectez-le à un nombre aussi réduit que possible de sources, telles que des périphériques. Ne sélectionnez pas **Affecter à tous les périphériques** lors de l'affectation du déclencheur et évitez d'affecter le déclencheur à de grands groupes de périphériques.

Débogage uniquement lors des tests

Le débogage ne doit être activé que lorsque vous travaillez activement sur votre déclencheur. Le débogage est utile pour tester que le déclencheur s'exécute et collecte les données attendues ; cependant, le débogage est une ponction inutile sur les ressources et doit être évité dans un environnement de production.

Rédiger un script de déclenchement optimisé

Les sections suivantes traitent des meilleures pratiques générales de codage ainsi que des lignes directrices relatives à l'utilisation des composants et des objets de l'API, tels que les tables et les enregistrements de session.

Application des meilleures pratiques de codage

Lors de la rédaction d'un script de déclenchement, nous recommandons les meilleures pratiques de codage suivantes afin d'optimiser et d'améliorer les performances des déclencheurs.

Corriger immédiatement les exceptions et les erreurs des déclencheurs

Les exceptions affectent gravement les performances. Si des exceptions ou des erreurs apparaissent dans le journal de débogage d'un déclencheur, corrigez ces problèmes immédiatement. Si certaines exceptions ne peuvent pas être facilement évitées, vous pouvez tenter de gérer l'exception à l'aide d'une instruction `try/catch`. L'instruction `try/catch` doit être intégrée dans une petite fonction qui n'effectue aucune autre opération

. Dans l'exemple ci-dessous, la fonction `JSON.parse` génère une exception si l'entrée n'est pas dans une syntaxe JSON bien formée. Comme il n'existe aucun moyen de valider que la syntaxe JSON est bien formée avant l'analyse, cette tâche est enveloppée dans une petite fonction d'aide.

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

Conserver les actions là où elles sont nécessaires dans le script

Déplacer les actions, telles que les opérations sur les chaînes de caractères et l'accès aux propriétés, dans la branche la plus exclusive qui nécessite l'action

```
.// inefficace : La variable cookies peut ne pas être nécessaire let
 cookies = HTTP.cookies ; if (HTTP.method === 'POST') { // traiter les
 cookies } // optimisé : Les cookies ne sont pas accédés à moins qu'ils
 ne soient nécessaires if (HTTP.method === 'POST') { let cookies =
 HTTP.cookies ; // traiter les cookies }
```

Stocker localement les objets souvent accédés

Conservez les cycles de l'unité centrale en stockant localement les applications, les géolocalisations, les appareils et les autres objets auxquels on accède plusieurs fois

```
// stockez l'objet localement let app = Application('exemple') ;
 app.commit() ; app.metricAddCount('custom', 1) ;
```

Éviter les variables dans la déclaration de débogage

N'incluez pas de variable dans la déclaration de débogage uniquement à des fins de débogage lorsque vous accédez à une charge utile ou à un autre objet volumineux. L'exclusion des variables de la déclaration de débogage vous permet de commenter la ligne de débogage et de conserver l'accès aux informations relatives à la charge utile.

Ne pas inclure la fonction exit() dans le script

Insérez une instruction `return` au lieu de la fonction globale `exit()`, qui est obsolète en raison de ses performances médiocres

```
// à éviter : ralentit le déclencheur HTTP.uri || exit() ; //
 recommandé if (!HTTP.uri) return;
```

> **Note:** Le comportement de l'instruction `return` est différent lorsqu'elle se trouve à l'intérieur d'une fonction imbriquée ; dans une fonction imbriquée, l'instruction `return` ne quitte que la fonction en cours, et non l'ensemble du déclencheur

.

Ne pas inclure la fonction eval() dans le script

Les déclencheurs ne doivent pas tenter de générer dynamiquement du code au moment de l'exécution ; par conséquent, la fonction `eval()` n'est pas prise en charge par l'API des déclencheurs ExtraHop. L'inclusion de la fonction dans le déclencheur peut provoquer des erreurs d'exécution imprévisibles.

Donner la priorité aux expressions régulières

Appliquez la méthode `test()` avec des expressions régulières au lieu de la méthode `match()`, lorsque cela est possible. L'extraction d'une valeur à l'aide d'une expression régulière est souvent plus rapide que l'analyse complète du XML ou des chaînes de requête

```
.// recherche plus lente let user = HTTP.parseQuery(payload).user ; //
 recherche plus rapide let match = /user=(.* ?)\&/i.exec(payload) ; let
 user = match ? match[1] : null ;
```

Appliquer des opérateurs d'égalité stricts

Pour les comparaisons, appliquez autant que possible des opérateurs d'égalité stricte, tels que triple equals (===) au lieu de loose equality (==). Les opérateurs d'égalité stricte sont plus rapides car ils ne tentent pas de coercions de type, comme la conversion des adresses IP en chaînes de caractères avant de les comparer.

Ne pas inclure de variables non déclarées

Déclarez les variables avec les instructions `let`, `var` ou `const`; n'incluez jamais de variables non déclarées

```
.// inefficace : La variable "cookies" n'est jamais déclarée cookies
 = HTTP.cookies ; // optimisé : La variable "cookies" est déclarée let
 cookies = HTTP.cookies ;
```

Organiser le script avec des fonctions

Les fonctions sont utiles pour les performances et l'organisation du code. Définissez les fonctions au début du déclencheur si votre déclencheur effectue la même opération à différents endroits du script de déclenchement, ou si les opérations du déclencheur peuvent être logiquement séparées en parties distinctes.

Organiser les fonctions d'aide au début du script

Définissez les fonctions d'aide en haut du script de déclenchement, plutôt qu'en bas, pour améliorer la lisibilité et aider le système ExtraHop à mieux comprendre et exécuter votre code.

Ajout de tables de session

La table de session partage globalement des types de données simples (tels que des chaînes) entre tous les déclencheurs et flux. Respectez les meilleures pratiques suivantes lorsque vous ajoutez des tables de session à votre script de déclenchement.

Incrémenter les comptes dans les tables de session

Lors du comptage, appelez la fonction `Session.increment` au lieu des fonctions `Session.lookup` ou `Session.replace`. La fonction `Session.increment` est atomique et son exactitude est donc garantie sur plusieurs threads de déclenchement.

Ne pas collecter de métriques de comptage distinctes à l'aide d'une table de session

Une table de session n'est pas un outil optimal pour compter le nombre de valeurs uniques, telles que les adresses IP, les ports ou les noms d'utilisateur, et est susceptible de causer des problèmes de performance. Au lieu de cela, engagez une métrique de comptage distinctpersonnalisée avec l'une des méthodes suivantes

```
:metricAddDistinct ( name : String, item : String|Number|IPAddress )
 metricAddDetailDistinct ( name : String, key : String|IPAddress,
 item : Chaîne|Nombre|AdresseIP )
```

Ne pas créer ou itérer de très grands tableaux ou objets

L'ajout de tableaux ou d'objets volumineux à une table de session peut réduire les performances. Par exemple, n'incluez pas plusieurs clés expirant, qui ont une valeur `notify=true`, dans un déclencheur qui s'exécute sur l'événement `SESSION_EXPIRE`. En outre, n'appelez pas les méthodes `JSON.parse` ou `JSON.stringify` sur des objets volumineux de la table de session.

Placer des valeurs dans la mémoire des flux

La mémoire de flux collecte les valeurs des événements d'un flux unique, comme une paire requête/réponse. Respectez les meilleures pratiques suivantes lorsque vous placez des valeurs dans la mémoire de flux dans un script de déclenchement.

Déterminer si des valeurs sont déjà disponibles

Avant de travailler avec la propriété `Flow.store`, vérifiez si les valeurs que vous souhaitez sont déjà disponibles directement à partir de l'API de déclenchement ExtraHop. Les versions récentes du micrologiciel ont supprimé la nécessité d'un magasin de flux pour un certain nombre de propriétés de demande qui étaient couramment consommées dans la réponse, telles que `HTTP.query` et `DB.statement`.

N'ajoutez que les valeurs nécessaires à la mémoire de flux

Évitez d'ajouter des objets volumineux à la propriété `Flow.store` si vous n'avez besoin que d'un sous-ensemble des valeurs de l'objet. Par exemple, le code suivant stocke uniquement la valeur de la propriété `TTL` de l'objet `DNS.answers`, au lieu de stocker toutes les valeurs de la propriété de l'objet `DNS.answers`

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

Effacer les valeurs lorsqu'elles ne sont plus nécessaires

Effacez les valeurs de la propriété `Flow.store` lorsqu'elles ne sont plus nécessaires en définissant la valeur de la propriété sur `null`. L'effacement de la mémoire de flux permet d'éviter les erreurs dans lesquelles le déclencheur traite la même valeur plusieurs fois.

Ne partagez pas les valeurs de la mémoire des flux entre les déclencheurs

N'accédez pas à l'objet `Flow.store` pour partager des valeurs entre différents déclencheurs s'exécutant sur le même événement. L'ordre dans lequel les déclencheurs s'exécutent lorsqu'un événement se produit n'est pas garanti. Si un déclencheur dépend de la valeur de la propriété `Flow.store` d'un second déclencheur, la valeur peut ne pas être conforme aux attentes et produire des résultats incorrects dans le premier déclencheur.

Création d'enregistrements personnalisés

Les mesures personnalisées permettent de capturer les données dont vous avez besoin si elles ne sont pas déjà fournies par les mesures de protocole intégrées au système ExtraHop. Respectez les meilleures pratiques suivantes lorsque vous créez des mesures personnalisées à l'aide d'un déclencheur.

Ne pas générer dynamiquement les noms des mesures personnalisées

La génération dynamique de noms de métriques personnalisés dégrade les performances et peut empêcher la découverte automatique des métriques dans le catalogue de métriques.

Ne pas convertir les adresses IP en chaîne avant de les ajouter aux mesures personnalisées

Si vous convertissez une adresse IP en chaîne avant de l'ajouter comme détail dans une mesure personnalisée, le système ExtraHop ne peut pas récupérer les attributs associés à l'adresse IP, tels que le périphérique ou le nom d'hôte associé.

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

Stockage des données dans des enregistrements

Les enregistrements vous permettent de stocker et d'extraire des informations structurées sur les flux de transactions, de messages et de réseaux. Les meilleures pratiques suivantes doivent être respectées lors de la création et du stockage d'enregistrements à l'aide d'un déclencheur.

Limiter l'accès aux propriétés via l'objet enregistrement

Évitez d'accéder aux propriétés par l'intermédiaire d'un objet d'enregistrement, tel que `HTTP.record`, si les propriétés sont disponibles dans l'objet protocole ou l'objet flux. Lorsque vous accédez à l'objet enregistrement, les données de l'enregistrement sont allouées en mémoire. Accédez plutôt à l'objet record lorsque vous stockez des données dans un enregistrement personnalisé modifié

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

Ne pas convertir les adresses IP en chaînes de caractères avant de les stocker dans les enregistrements

Le système ExtraHop filtre et trie les adresses IP, ce qui ne peut pas être appliqué à des chaînes telles que les blocs CIDR pour les adresses IPv4.

Accès au magasin de données

Les déclencheurs du magasin de données vous permettent d'accéder à des mesures agrégées qui ne sont pas disponibles pour les événements basés sur des transactions tels que `HTTP_REQUEST`. Respectez les meilleures pratiques suivantes lorsque vous accédez au magasin de données par le biais d'un déclencheur.

Créer des déclencheurs de magasin de données avec parcimonie

Les ressources des déclencheurs du magasin de données sont très limitées. Dans la mesure du possible, évitez d'exécuter des déclencheurs sur les événements suivants du magasin de données :

- ALERT_RECORD_COMMIT
- METRIC_RECORD_COMMIT
- METRIC_CYCLE_BEGIN
- METRIC_CYCLE_END

Par exemple, si vous ne souhaitez accéder à une table de session que périodiquement, exécutez le déclencheur sur l'événement `SESSION_EXPIRE` au lieu de l'événement `METRIC_CYCLE_END`

.

Configurer les options avancées lors de la validation des enregistrements métriques

Assurez-vous que les options avancées des déclencheurs sont configurées pour les déclencheurs qui s'exécutent sur l'événement `METRIC_RECORD_COMMIT`. En particulier, nous vous recommandons de configurer un cycle métrique de 30 secondes pour les déclencheurs qui s'exécutent sur l'événement `METRIC_RECORD_COMMIT`.

Évaluer les performances des déclencheurs

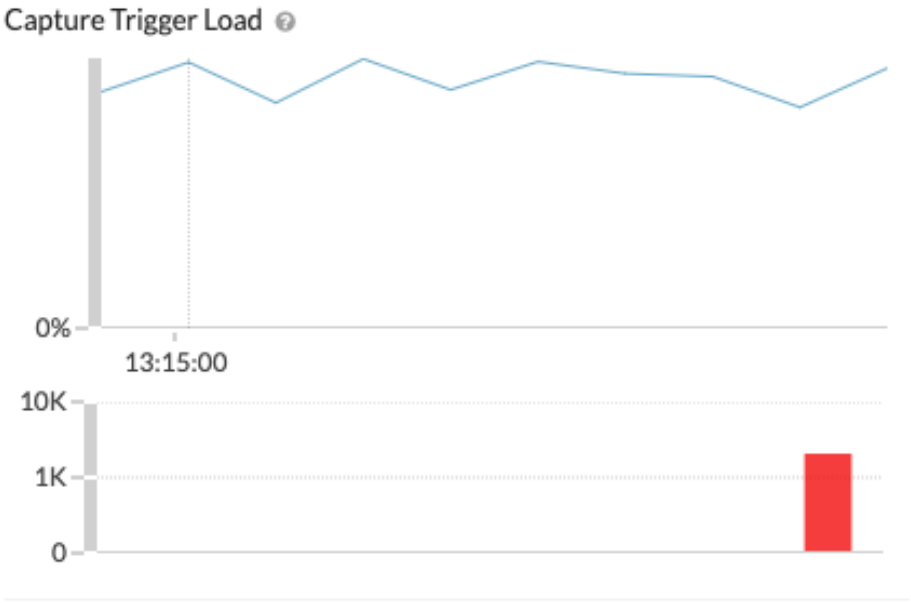
Vous pouvez surveiller et évaluer l'impact des déclencheurs exécutés sur votre système ExtraHop de plusieurs façons. Vous pouvez afficher des informations sur les performances d'un déclencheur individuel et plusieurs graphiques indiquant l'impact collectif de tous vos déclencheurs sur le système.

Afficher les performances d'un déclencheur individuel

Vous pouvez vérifier si un déclencheur présente des erreurs ou des exceptions à partir de l'onglet Debug Log (journal de débogage) du volet Edit Trigger (modifier le déclencheur). Accédez à la page Triggers en cliquant sur l'icône System Settings (Paramètres système) dans le système ExtraHop.

Vous pouvez afficher le coût des performances d'un déclencheur en cours d'exécution sur le graphique Capture Trigger Load (Chargement du déclencheur de capture) dans l'onglet Edit Trigger (Modifier le déclencheur). Le tableau affiche un graphique des performances du déclencheur qui indique le nombre de

cycles consommés par le déclencheur dans un intervalle de temps donné. Vous pouvez survoler un point de données pour afficher les principales mesures de performance à un moment donné.



L'astuce de survol comprend les informations suivantes :

- Le plus grand et le plus petit nombre de cycles consommés par le déclencheur pour traiter un seul événement.
- Le nombre de fois où le déclencheur s'est exécuté et le pourcentage de fois où il s'est exécuté par rapport à tous les déclencheurs qui se sont exécutés dans la même plage de temps.
- Le nombre total de cycles consommés par le déclencheur et le pourcentage de cycles consommés par rapport à tous les déclencheurs exécutés dans la même plage horaire.

Afficher les performances de tous les déclencheurs du système

La page Santé du système contient plusieurs graphiques qui donnent un aperçu des déclencheurs en cours d'exécution sur le système ExtraHop. Accédez à la page Santé du système en cliquant sur l'icône Paramètres du système dans le système ExtraHop.

En consultant les graphiques de santé du système décrits dans les directives suivantes, vous pouvez surveiller les problèmes susceptibles d'affecter les performances du système ou d'entraîner des données incorrectes.

Vérifier que le déclencheur est en cours d'exécution

Le tableau Détails des déclencheurs affiche tous les déclencheurs en cours d'exécution sur le système. Si le déclencheur que vous venez de créer ou de modifier n'est pas répertorié, il se peut que le script du déclencheur présente un problème.

Surveillez les activités inattendues

Le graphique Déclencheurs exécutés et abandonnés peut afficher des rafales d'activité des déclencheurs qui peuvent indiquer un comportement inefficace d'un ou de plusieurs déclencheurs. Si des rafales sont affichées, consultez le graphique Exécution des déclencheurs par déclencheur pour localiser tout déclencheur consommant plus de ressources que la moyenne, ce qui peut indiquer que le déclencheur a un script mal optimisé qui affecte les performances.

Vérifier les exceptions de déclenchement non gérées

Le graphique Exceptions par déclencheur affiche toutes les exceptions causées par les déclencheurs. Les exceptions contribuent largement aux problèmes de performances du système et doivent être corrigées immédiatement.

Vérifier les déclencheurs retirés de la file d'attente

Le graphique Déclencheurs exécutés et abandonnés affiche le nombre de déclencheurs qui ont été retirés de la file d'attente des déclencheurs. Une cause fréquente d'abandon de déclencheurs est un déclencheur de longue durée qui domine la consommation de ressources. Un système sain ne devrait jamais avoir de déclenchements abandonnés.

Surveiller la consommation des ressources

Le graphique Charge de déclenchement permet de suivre l'utilisation de toutes les ressources disponibles par les déclencheurs. Une charge élevée correspond à environ 50 %. Recherchez les pics de consommation qui peuvent indiquer qu'un nouveau déclencheur a été introduit ou qu'un déclencheur existant rencontre des problèmes.

Vous pouvez vérifier si vos déclencheurs de magasin de données, également appelés déclencheurs de pont, fonctionnent correctement à l'aide des graphiques suivants :

- Exécutions et abandons de déclencheurs de magasin de données
- Charge de déclenchement du magasin de données
- Exceptions des déclencheurs de magasin de données par déclencheur

Consultez la FAQ sur la santé du système pour prendre connaissance des questions fréquemment posées sur la façon dont les graphiques de santé du système peuvent vous aider à évaluer les performances des déclencheurs sur votre système ExtraHop.

Voir les ressources sur les déclencheurs

Pour en savoir plus sur les déclencheurs et l'API ExtraHop Trigger, consultez les ressources suivantes :

Documentation

- Référence API ExtraHop Trigger
- SectionDémarrer avec les déclencheurs du Guide de l'interface Web.

Exemples d'application

- Walkthrough sur les déclencheurs : Créer un déclencheur pour collecter des mesures personnalisées pour les erreurs HTTP 404
- UPA walkthrough : Construire un déclencheur pour surveiller les réponses aux requêtes NTP monlist
- Découverte d'un flux de données ouvert : Configurer un flux de données ouvert pour envoyer des données métriques à AWS Cloudwatch

Formation

- Planification d'un déclencheur
- Création d'un déclencheur simple
- Créer et utiliser des conteneurs d'applications
- Création d'un déclencheur multi-événements

Communauté

- Forums de la communauté ExtraHop