Exporter les journaux des interactions API du service d'apprentissage automatique

Publié: 2023-09-19

Vous pouvez configurer les capteurs et les consoles pour qu'ils exportent les journaux des interactions API avec le service d'apprentissage automatique ExtraHop. Le système ExtraHop exporte les journaux de l'API par le biais de requêtes HTTPS POST. N'importe quel serveur HTTP peut recevoir les journaux, à condition que le serveur soit accessible par le système ExtraHop et qu'un certificat TLS soit installé sur le serveur.

Configuration d'un serveur HTTP pour recevoir les journaux

Avant de configurer le système ExtraHop pour qu'il exporte les journaux d'API, vous devez configurer un serveur HTTP pour qu'il reçoive et enregistre les journaux. Cette rubrique explique comment configurer un exemple d'application Go disponible dans le dépôt GitHub ExtraHop code-examples.

Avant de commencer

- Vous devez installer Go sur votre machine. Pour plus d'informations, consultez la documentation Go à l'adresse https://go.dev/learn/.

1. Générez un certificat de serveur pour le serveur HTTP.

   Note: Si vous disposez déjà d'un certificat, vous pouvez sauter cette étape.

   a) Exécutez la commande suivante pour générer la clé de l'autorité de certification (CA) :

   ```
   openssl genrsa -aes256 -out serverca.key 4096
   ```

   b) Exécutez la commande suivante pour générer le certificat de l'autorité de certification :

   ```
   openssl req -new -key serverca.key -x509 -out serverca.crt -days 3650
   ```

   c) Exécutez la commande suivante avec vos variables pour générer une demande de signature de certificat :

   ```
   openssl req -new \
       -nodes \
       -newkey rsa:4096 \
       -keyout server.key \
       -out server.req \
       -batch \
       -subj "/C=US/ST=WA/L=Seattle/O=ORGANIZATION_NAME/OU=router/
   CN=SERVER_URL" \
       -reqexts SAN \
       -config <(cat /etc/ssl/openssl.cnf <(printf
    "[SAN]\nsubjectAltName=DNS:SERVER_URL,IP:SERVER_IP_ADDRESS"))
   ```

   Remplacez les variables suivantes dans la commande ci-dessus :

   - **ADRESSE_IP_DU_SERVEUR:** L'adresse IP de votre serveur.
   - **SERVER_URL:** L'URL de votre serveur.
   - **NOM_ORGANISATION:** Le nom de votre organisation.

   d) Exécutez la commande suivante avec vos variables pour générer le certificat du serveur :

   ```
   openssl x509 -req \
       -in server.req \
       -CA serverca.crt \
   ```

```
    -CAkey serverca.key \
    -CAcreateserial \
    -out server.crt \
    -days 3650 \
    -sha256 \
    -extfile <(printf
"subjectAltName=DNS:SERVER_URL,IP:SERVER_IP_ADDRESS")
```

Remplacez les variables suivantes dans la commande ci-dessus :

- **ADRESSE_IP_DU_SERVEUR:** L'adresse IP de votre serveur.
- **SERVER_URL:** L'URL de votre serveur.

e) Exécutez la commande suivante pour créer un certificat dans un format que vous pouvez télécharger vers le capteur ou la console.

```
cat server.key server.crt serverca.crt > trusted-server.pem
```

2. Allez sur le dépôt GitHub ExtraHop code-examples et téléchargez le fichier `ml_api_logger/ml_api_logger.go` sur votre machine locale.
3. Exécutez la commande suivante, en remplaçant `CERT_PATH` par le chemin du certificat TLS que vous avez généré pour le serveur :

```
export LOGGER_CERT=CERT_PATH
```

4. Exécutez la commande suivante, en remplaçant `KEY_PATH` par le chemin de la clé avec laquelle vous avez signé le certificat :

```
export LOGGER_KEY=KEY_PATH
```

5. Exécutez la commande suivante, en remplaçant PORT par le port sur lequel le serveur écoute :

```
export LOGGER_PORT=PORT
```

6. Dans le répertoire ml_api_logger, exécutez la commande suivante pour compiler le code Go :

```
go build ml_api_logger.go
```

7. Exécutez la commande suivante pour démarrer le serveur et enregistrer les journaux de l'API à l'adresse `extrahop-ade-log.json`:

```
ml_api_logger > extrahop-ade-log.json
```

Configurer le capteur ou la console

Vous devez configurer le capteur ou la console pour exporter les journaux vers le serveur que vous avez configuré.

- Si le certificat de votre serveur n'est pas approuvé par le certificat intégré au capteur, vous devez ajouter le certificat au capteur ou à la console.

1. Connectez-vous aux paramètres d'administration du capteur ou de la console via `https://<extrahop-hostname-or-IP-address>/admin`.
2. Dans la section Appliance Settings (Paramètres de l'appareil), cliquez sur **Running Config (Configuration en cours d'exécution)**.
3. Cliquez sur **Edit config**.
4. Dans la section `hop_cloud`, ajoutez une entrée dont la clé est `api_logging_target` et la valeur est un objet avec les champs suivants :

enabled : *Booléen*

Spécifie si la journalisation des interactions API est activée. Spécifiez `true`.

hostname : *Chaîne*

Le nom d'hôte du serveur que vous avez configuré pour recevoir les journaux d'interaction avec l'API.

port : *Nombre*

Le port sur lequel le serveur tiers écoute.

La section `hop_cloud` mise à jour doit ressembler au JSON suivant :

```
  "hopcloud" : {"api_logging_target" : {"enabled" : true "hostname" :
"example.extrahop.com" "port" : 100 } "analysis_settings" : {} }
```

Format du journal de l'API

Les journaux sont exportés au format JSON. Chaque journal d'une requête HTTPS vers le service d'apprentissage automatique contient les champs suivants :

séquence : *Numéro*

Un identifiant numérique qui met en corrélation les demandes et les réponses. Par exemple, si une demande a un numéro de séquence de 1, le journal de la réponse aura également un numéro de séquence de 1.

request : *Objet*

Un objet qui contient des détails sur la demande. L'objet contient les champs suivants :

Close : *Booléen*

Indique si l'en-tête Connection est défini sur close.

ContentLength : *Nombre*

Valeur de l'en-tête ContentLength.

En-tête : *Objet*

Un objet qui contient les en-têtes HTTPS.

Host : *Chaîne*

Le nom d'hôte du serveur.

Method (Méthode) : *Chaîne*

La méthode de la requête.

Proto : *Chaîne*

Le protocole HTTP avec lequel la demande a été envoyée.

RemoteAddr : *Chaîne*

L'adresse IP du serveur.

RequestURI : *Chaîne*

L'URI de la requête.

TLSVersion : *Chaîne*

La version TLS avec laquelle la demande a été cryptée.

Trailer : *Chaîne*

La valeur de l'en-tête Trailer.

TransferEncoding : *Chaîne*

La valeur de l'en-tête Transfer-Encoding.

request_body : *Objet*

Le corps JSON des requêtes POST, PUT et PATCH.

Chaque journal d'une réponse HTTPS du service d'apprentissage automatique contient les champs suivants :

séquence : *Nombre*

Un identifiant numérique qui établit une corrélation entre les demandes et les réponses. Par exemple, si une demande a un numéro de séquence de 1, le journal de la réponse aura également un numéro de séquence de 1.

response_status_code : *Nombre*

Code d'état de la réponse.

response_headers : *Objet*

Objet contenant les en-têtes HTTPS.

response_body : *Objet*

Le corps JSON de la réponse.

Exemple de demande

L'objet JSON suivant est un exemple de journal pour une demande d'API :

```
{"sequence" : 302, "request" : { "Method" : "POST", "Host" :
 "appliance.example.extrahop.com", "RemoteAddr" : "127.0.0.1:1234",
 "RequestURI" : "/api/v1/metrics", "TLSVersion" : "TLS1.2", "Proto" :
 "HTTP/1.1", "ContentLength" : 149, "TransferEncoding" : null, "Header" :
 {"Accept" : [ "application/json" ], "Content-Length" : ["149" ], "Content-
Type" : [ "application/json" ] }, "Close" : false, "Trailer" : null },
 "request_body" : {"metric_category" : "net", "from" : -1, "object_type" :
 "capture", "object_ids" : [ 0 ], "until" : 0, "metric_specs" : [ { "name" :
 "pkts" } ], "cycle" : "30sec" } }
```

Exemple de réponse

L'objet JSON suivant est un exemple de journal pour une réponse d'API :

```
{"sequence" : 302, "response_status_code" : "200", "response_headers" :
 {"Content-Type" : ["application/json ; charset=utf-8" ], "Vary" : ["Accept-
Encoding" ] }, "response_body" : {"cycle" : "30sec", "node_id" : 0,
 "clock" : 1678150650000, "from" : 1678150649999, "until" : 1678150650000,
 "stats" : [ { "oid" : 0, "time" : 1678150650000, "duration" : 30000,
 "values" : [ 1260 ] } ] } }
```