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

Publié: 2024-11-04

Vous pouvez configurer des capteurs et des consoles pour exporter les journaux des interactions des API avec le service d'apprentissage automatique ExtraHop. Le système ExtraHop exporte les journaux d'API via des requêtes HTTPS POST. N'importe quel serveur HTTP peut recevoir les journaux, à condition qu'il 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 exporter les journaux d'API, vous devez configurer un serveur HTTP pour recevoir et enregistrer les journaux. Cette rubrique explique comment configurer un exemple d'application Go disponible dans Référentiel GitHub d'exemples de code ExtraHop.

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 possédez déjà un certificat, vous pouvez ignorer 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 CA :

   ```
   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.
   - **URL DU SERVEUR:** URL de votre serveur.
   - **NOM_ORGANISATION:** Le nom de votre organisation.

d) Exécutez la commande suivante avec vos variables pour générer le certificat de 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.
- **URL DU SERVEUR:** URL de votre serveur.

e) Exécutez la commande suivante pour créer un certificat dans un format que vous pouvez télécharger sur la sonde ou la console.

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

2. Accédez au Référentiel GitHub d'exemples de code ExtraHop et téléchargez le `ml_api_logger/ml_api_logger.go` fichier sur votre machine locale.
3. Exécutez la commande suivante en remplaçant `CERT_PATH` avec 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` avec 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. Limitez l'enregistreur pour qu'il ne reçoive des paquets que sur une adresse IP de serveur spécifique.

   Par défaut, l'application Go reçoit des journaux sur toutes les adresses IP configurées pour le serveur. Pour limiter l'application à une seule adresse IP, exécutez la commande suivante en remplaçant IP par l'adresse IP :

```
export LOGGER_IP=IP
```

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

```
go build ml_api_logger.go
```

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

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

Configuration de la sonde ou de la console

Vous devez configurer la sonde 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 à la sonde, vous devez ajouter le certificat à la sonde ou à la console.

1. Connectez-vous aux paramètres d'administration de la sonde ou de la console via `https://<extrahop-hostname-or-IP-address>/admin`.
2. Dans la section Paramètres de l'appliance, cliquez sur **Configuration en cours d'exécution**.
3. Cliquez **Modifier la configuration**.
4. Dans le `hop_cloud` section, ajoutez une entrée où la clé est `api_logging_target` et la valeur est un objet avec les champs suivants :

   activé : *Booléen*

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

   nom d'hôte : *Corde*

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

   port : *Numéro*

   Port sur lequel le serveur tiers écoute.

   Le mis à jour `hop_cloud` la section 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 adressée au service d'apprentissage automatique contient les champs suivants :

séquence : *Numéro*

Identifiant numérique qui met en corrélation les demandes et les réponses. Par exemple, si une demande possède un numéro de séquence de 1, le journal des réponses aura également un numéro de séquence de 1.

demande : *Objet*

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

Fermer : *Booléen*

Indique si l'en-tête Connection est configuré pour se fermer.

Longueur du contenu : *Numéro*

Valeur de l'en-tête ContentLength.

En-tête : *Objet*

Objet contenant les en-têtes HTTPS.

Hôte : *Corde*

Le nom d'hôte du serveur.

Méthode : *Corde*

Méthode de la demande.

Porto : *Corde*

Le protocole HTTP avec lequel la demande a été envoyée.

Adresse de la télécommande : ***Corde***

L'adresse IP du serveur.

URI de la demande : ***Corde***

L'URI de la demande.

Version TLS : ***Corde***

Version TLS avec laquelle la demande a été cryptée.

Bande-annonce : ***Corde***

La valeur de l'en-tête de la bande-annonce.

Encodage du transfert : ***Corde***

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

corps de la demande : ***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 : ***Numéro***

Identifiant numérique qui met en corrélation les demandes et les réponses. Par exemple, si une demande possède un numéro de séquence de 1, le journal des réponses aura également un numéro de séquence de 1.

code_état_de la réponse : ***Numéro***

Le code dac.state de la réponse.

en-têtes de réponse : ***Objet***

Un objet contenant les en-têtes HTTPS

corps_réponse : ***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
                ]
            }
        ]
    }
}
```