

# Création d'une détection personnalisée

Publié: 2024-01-31

Les détections personnalisées vous permettent de spécifier des critères qui génèrent des détections sur le système ExtraHop. L'apprentissage automatique et les détections basées sur des règles détectent les comportements inhabituels et les menaces courantes. Toutefois, en créant une détection personnalisée, vous pouvez cibler les appareils et les comportements essentiels pour votre réseau.

Lorsque vous créez une détection personnalisée, vous devez créer un déclencheur qui identifie l'événement système et les conditions que le système doit surveiller, puis vous pouvez attribuer le déclencheur aux appareils ou groupes d'équipements spécifiques que vous souhaitez surveiller. Lorsque l'événement se produit, une détection est générée.

Dans ce guide, nous fournissons les étapes et un exemple de script qui génère une détection personnalisée lorsque des connexions suspectes sont établies avec des sites Web spécifiques via Windows PowerShell.

## Avant de commencer

- Vous devez avoir une certaine connaissance d'ExtraHop [déclencheurs](#). En particulier, considérez [ces meilleures pratiques](#) lors de l'écriture de votre script et de l'attribution de déclencheurs.
- Vous devez disposer d'un compte utilisateur auprès du [privilèges](#) nécessaire pour créer des déclencheurs.
- Si vous avez un console, créez un déclencheur sur le console et le déclencheur fonctionnera sur tous les capteurs connectés.


## Créez un déclencheur pour générer des détections personnalisées


Les déclencheurs génèrent des détections personnalisées en appelant le `commitDetection` fonction dans le script du déclencheur.

Dans l'exemple suivant, le déclencheur génère une détection personnalisée lorsqu'un client PowerShell accède à un site Web connu sous le nom de site intermédiaire pour les données exfiltrées.

Le déclencheur identifie les connexions PowerShell en recherchant les hachages JA3 du client SSL appartenant à des clients PowerShell connus.

Si la connexion SSL est établie entre un client PowerShell et un hôte suspect, le déclencheur génère une détection. La détection inclut la version de PowerShell qui a initié la connexion, l'adresse IP du serveur et l'adresse IP du client.

 **Note:** Pour plus d'informations sur le `commitDetection` fonction, voir [Référence de l'API Trigger](#)

1. Cliquez sur l'icône des paramètres système  puis cliquez sur **DÉCLENCHEURS**.
2. Cliquez **Créez**.
3. Spécifiez les paramètres de configuration du déclencheur suivants :

### Nom

Tapez le nom de votre déclencheur. Ce nom identifie votre déclencheur, et non la détection.

Dans notre exemple, nous allons entrer le nom : `Détection personnalisée : connexion PowerShell à un site suspect`.

### Descriptif

(Facultatif) Entrez la description du déclencheur. Cette description concerne le déclencheur et non la détection.

Dans notre exemple, nous allons saisir la description : `Crée une détection chaque fois qu'un client PowerShell se connecte à pastebin,`

raw.githubusercontent.com ou githack. Les clients PowerShell sont identifiés par des hachages JA3.

## Évènements

Sélectionnez l'événement sur lequel le déclencheur s'exécute.

Dans notre exemple, nous allons sélectionner l'événement SSL\_OPEN. Cet événement se produit lorsqu'une connexion SSL est établie pour la première fois.

## Missions

Sélectionnez l'équipement ou le groupe d'équipements que vous souhaitez surveiller. Dans un premier temps, attribuez votre déclencheur à un seul équipement à des fins de test. Après avoir vérifié que la détection personnalisée fonctionne correctement, assignez le déclencheur à un groupe d'équipements contenant tous les appareils que vous souhaitez surveiller.

PowerShell étant un outil de ligne de commande Windows, sélectionnez un serveur Microsoft pour tester le déclencheur. Après avoir confirmé que la détection personnalisée fonctionne correctement, modifiez l'attribution à un groupe d'équipements contenant tous vos serveurs Microsoft critiques. Pour plus d'informations sur la création de groupes d'équipements, voir [Création d'un groupe d'équipements](#).

4. Dans le volet droit, tapez le code qui détermine le moment où votre détection personnalisée est générée.

Dans notre exemple, le code déclencheur suivant identifie le moment où un client établit une connexion à pastebin, githubusercontent ou githack :

```
if(SSL.host.match(/pastebin/i) || SSL.host.match(/
raw.githubusercontent.com/i) || SSL.host.match(/githack/i)) {
}
}
```

5. Tapez ensuite le code qui valide votre détection personnalisée. Le `commitDetection` la fonction doit être écrite dans le format suivant :

```
commitDetection('<detection type ID>', {
  title: '<title>',
  description: '<detection description>',
  categories: ['<category>'],
  riskScore: <risk score>,
  participants: [{
    object:<offender participant>,
    role: 'offender'
  }, {
    object: <victim participant>,
    role: 'victim'
  }],
  identityKey: '<identity key>',
  identityTtl: '<time period>',
});
```

Entrez des valeurs pour chacun des paramètres suivants dans votre script.

Valeur	Descriptif
ID du type de détection	Chaîne unique qui identifie votre détection personnalisée. Cette chaîne ne peut contenir que des lettres, des chiffres et des traits de soulignement.
titre	Texte qui apparaît en haut de la carte de détection. Tapez un titre descriptif facile à scanner.

Valeur	Descriptif
description de la détection	<p>Ce titre apparaît dans le catalogue de détection en tant que nom d'affichage de votre type de détection, précédé de [personnalisé].</p> <p>Texte qui apparaît sous le titre et la catégorie sur une carte de détection. Entrez les informations relatives à l'événement à l'origine de la détection.</p> <p>Ce champ prend en charge le markdown. Nous vous recommandons d'inclure des variables d'interpolation pour afficher des informations spécifiques sur votre détection.</p> <p>Par exemple, les variables <code>\$(Flow.client.ipaddr)</code> et <code>\$(Flow.server.ipaddr)</code> affichent l'adresse IP du client et de l'équipement serveur dans le flux et <code>\$(Flow.l7proto)</code> affiche le protocole L7. Inclure <code>\n</code> à la fin de chaque ligne de texte pour s'assurer que la description s'affiche correctement.</p>
indice de risque	<p>Un chiffre qui mesure la probabilité, la complexité et l'impact commercial d'une détection de sécurité. L'icône de l'indice de risque apparaît en haut de la carte de détection et est codée par couleur en fonction de la gravité, en rouge (80-99), orange (31-79) ou jaune (1-30). Tu peux <a href="#">trier les détections par risque</a>.</p>
délinquant participant victime participant	<p>Ensemble d'objets identifiant les participants à la détection. Définissez le rôle du participant comme suit : 'offender' ou 'victim' et fournissez une référence à un équipement, à une adresse IP ou à un objet d'application pour ce rôle.</p> <p>Par exemple, le tableau suivant identifie le serveur en tant que délinquant et le client en tant que victime dans un flux :</p> <pre data-bbox="878 1419 1458 1646"> participants: [   { role: 'offender', object:     Flow.server.device},   { role: 'victim', object:     Flow.client.device } ]</pre>
clé d'identité	<p>Chaîne qui permet d'identifier les détections en cours. Si plusieurs détections avec la même clé d'identité et le même type de détection sont générées dans le délai spécifié par le</p>

Valeur	Descriptif
	<p>identityTtl paramètre, les détections sont consolidées en une seule détection continue.</p> <p>Créez une chaîne de clé d'identité unique en combinant les caractéristiques de la détection.</p> <p>Par exemple, la clé d'identité suivante est créée en combinant l'adresse IP du serveur et l' adresse IP du client :</p> <pre data-bbox="876 472 1451 577">identityKey: [Flow.server.ipaddr, Flow.client.ipaddr].join('!!!')</pre>
période	<p>Durée après la génération d'une détection pendant laquelle les détections dupliquées sont consolidées dans une détection continue. La période est réinitialisée et la détection ne prend fin qu'à son expiration.</p> <p>Les périodes de validité suivantes sont les suivantes :</p> <ul data-bbox="876 871 998 976" style="list-style-type: none"> <li>• hour</li> <li>• day</li> <li>• week</li> </ul> <p>La période par défaut est hour.</p>

L'exemple suivant montre la section de script terminée.

```
commitDetection('powershell_ja3', {
  title:
'PowerShell / BitsAdmin Suspicious Connection',
  description:
"This SSL client matched a variant of PowerShell." + "\n"+
"Investigate other client behaviors on the victim host." + "\n"+
"- ** PowerShell/BitsAdmin JA3 client match**" + "\n"+
"- **Client IP:** " + Flow.client.ipaddr + "\n"+
"- **JA3 Client Value:** " + ja3 + "\n"+
"- **JA3 Client Match:** " + suspect_ja3_hashes[ja3],
  riskScore: 60,
  participants: [{
    object:Flow.client.device,
    role: 'offender'
  }],
  identityKey: [
    Flow.server.ipaddr,
    Flow.client.ipaddr,
    hash
  ].join('!!!'),
  identityTtl: 'hour',
});
```

Ces valeurs apparaissent sur la carte de détection de la même manière que dans la figure suivante :

The screenshot shows a detection alert interface. On the left, labels point to various parts of the alert: 'detection type ID' points to the title 'powershell\_ja3'; 'risk score' points to the '60 RISK CAUTION' indicator; 'category' points to the title; 'description' points to the text block containing technical details; and 'participants' points to the offender information box.

**detection type ID** — powershell\_ja3

**risk score** — 60 RISK CAUTION

**category** — powershell\_ja3

**description** — This SSL client matched a variant of PowerShell. Investigate other client behaviors on the victim host.  
 - \*\* PowerShell/BitsAdmin JA3 client match\*\*  
 - \*\*Client IP:\*\* 192.168.131.109  
 - \*\*JA3 Client Value:\*\* 8c4a22651d328568ec66382a84fc505f:BitsAdmin/PowerShell 5.0 Windows 7 64 bit enterprise  
 - \*\*JA3 Client Match:\*\* 8c4a22651d328568ec66382a84fc505f:BitsAdmin/PowerShell 5.0 Windows 7 64 bit enterprise

**participants** — OFFENDER  
 workstation05.example.com  
 192.168.131.109

Sep 16 10:43  
 lasting a few seconds

6. Cliquez **Enregistrer** puis cliquez sur **Terminé**.

Voir [Exemple de déclencheur de détection personnalisé](#) pour un script annoté complet.

Votre détection personnalisée sera ajoutée au catalogue de détection une fois que votre déclencheur sera exécuté pour la première fois. [Ajouter des catégories de détection et des techniques MITRE](#) à la détection depuis le catalogue de détection.

## Création d'un type de détection personnalisé

Après avoir créé un déclencheur pour générer votre détection personnalisée, vous pouvez créer un type de détection personnalisé dans le catalogue de détection pour ajouter des informations supplémentaires à votre détection.

Vous pouvez spécifier un nom d'affichage et ajouter des catégories de détection pour vous aider à localiser votre détection sur la page Detections. Vous pouvez également ajouter des liens MITRE, qui permettent à votre détection personnalisée d'apparaître dans la matrice de la page Grouper par technique MITRE.

1. Connectez-vous au système ExtraHop via `https://<extrahop-hostname-or-IP-address>`.
2. Cliquez sur l'icône des paramètres système puis cliquez sur **Catalogue de détection**.
3. Sur la page Catalogue de détection, effectuez l'une des étapes suivantes :
  - Si votre déclencheur est déjà lancé, le système ajoute automatiquement votre détection personnalisée au catalogue avec le nom d'affichage spécifié dans le déclencheur précédé de [personnalisé]. Cliquez sur le type de détection à modifier.
  - Si votre type de détection n'a pas été créé, cliquez sur **Créez**.
4. Renseignez les champs suivants :

### Nom d'affichage

Entrez un nom unique pour le titre de la détection.

### ID du type de détection

Tapez la valeur que vous avez saisie pour l'ID du type de détection dans le déclencheur. Par exemple, si vous avez saisi : `commitDetection('network_segmentation_breach')`, l'ID du type de détection est « network\_segmentation\_breach ». Vous ne pouvez pas modifier l'ID du type de détection une fois le type de détection enregistré.

### Auteur

Entrez l'auteur de la détection personnalisée.

### Technique MITRE

Dans la liste déroulante, sélectionnez une ou plusieurs techniques MITRE que vous souhaitez associer à la détection.

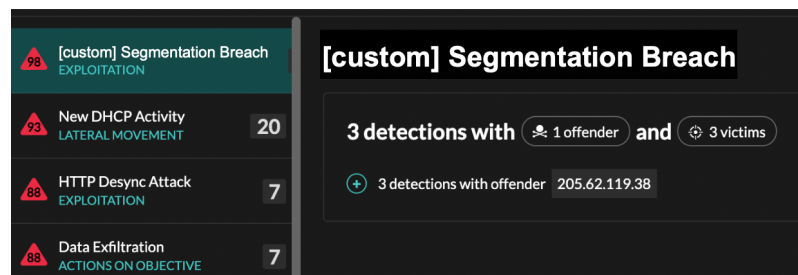
5. Cliquez **Enregistrer**.

## Afficher les détections personnalisées

Vous pouvez consulter les détections personnalisées sur le Détections page avec d'autres détections intégrées.

Regroupez la page des détections [par type](#). Toutes les détections de la liste de détection sont regroupées par type de détection.

Par exemple, si votre nom d'affichage de détection est `[custom]Segmentation Breach`, l'entrée apparaîtrait dans la liste de détection comme dans la figure suivante :



En haut à gauche de la page, sélectionnez **Carte MITRE**. Les techniques MITRE liées à la détection personnalisée sont mises en évidence dans la matrice.

### Prochaines étapes

[Création d'une règle de notification de détection](#). Par exemple, vous pouvez configurer le système ExtraHop pour qu'il vous envoie un e-mail lorsque votre détection personnalisée se produit.

## Exemple de déclencheur de détection personnalisé

Le script suivant est l'exemple complet de PowerShell/JA3 auquel il est fait référence dans ces instructions.

```
// If the server is internal, exit
if ( ! Flow.server.ipaddr.isExternal ) {
    return;
}
// If the SSL host name is not set, exit
if(SSL.host === null) { return; }

// Continue only if the SSL hostname belongs to one of the suspicious sites
if(SSL.host.match(/pastebin/i) || SSL.host.match(/raw.githubusercontent.com/i) || SSL.host.match(/githack/i)) {

    // List of common PowerShell JA3 hashes
    let suspect_ja3_hashes = cache('suspect_ja3_hashes', () => ( {
        '13cc575f247730d3eeb8ff01e76b245f': 'PowerShell/BitsAdmin/PowerShell
4.0 Windows Server 2012RT',
```

```

'5e12c14bda47ac941fc4e8e80d0e536f': 'PowerShell/BitsAdmin/PowerShell
4.0 Windows Server 2012RT',
'2c14bfb3f8a2067fbc88d8345e9f97f3': 'PowerShell/BitsAdmin Windows
Server 2012RT',
'613e01474d42ebe48ef52dff6a20f079': 'PowerShell/BitsAdmin Windows
Server 2012RT',
'05af1f5calb87cc9cc9b25185115607d': 'BitsAdmin/PowerShell 5.0 Windows
7 64 bit enterprise',
'8c4a22651d328568ec66382a84fc505f': 'BitsAdmin/PowerShell 5.0 Windows
7 64 bit enterprise',
'235a856727c14dba889ddee0a38dd2f2': 'BitsAdmin/PowerShell 5.1 Server
2016',
'17b69de9188f4c205a00fe5ae9c1151f': 'BitsAdmin/PowerShell 5.1 Server
2016',
'd0ec4b50a944b182fc10ff51f883ccf7': 'PowerShell/BitsAdmin (Microsoft
BITS/7.8) Server 2016',
'294b2f1dc22c6e6c3231d2fe311d504b': 'PowerShell/BitsAdmin (Microsoft
BITS/7.8) Server 2016',
'54328bd36c14bd82ddaa0c04b25ed9ad': 'BitsAdmin/PowerShell 5.1 Windows
10',
'fc54e0d16d9764783542f0146a98b300': 'BitsAdmin/PowerShell 5.1 Windows
10',
'2863b3a96f1b530bc4f5e52f66c79285': 'BitsAdmin/PowerShell 6.0 Windows
Server 2012RT',
'40177d2da2d0f3a9014e7c83bdeee15a': 'BitsAdmin/PowerShell 6.0 Windows
Server 2012RT',
'36f7277af969a6947a61ae0b815907a1': 'PowerShell/BitsAdmin Windows 7
32 bit enterprise',
    });
    // Store the client JA3 hash in a variable
    const hash = SSL.ja3Hash;

    // Iterate through each PowerShell JA3 hash
    for ( let ja3 in suspect_ja3_hashes ) {

        // If the client JA3 hash is from PowerShell,
        // commit the detection
        if ( hash.includes(ja3) ) {

            commitDetection('PowerShell_JA3', {
                categories: ['sec.caution'],
                title: "PowerShell / BitsAdmin Suspicious Connection",
                // Specify the offender as the device object of the client
                participants: [
                    { role: 'offender', object: Flow.client.device }
                ],
                description:
                    "This SSL client matched a variant of PowerShell." +
"\n"+
                    "Investigate other client behaviors on the victim host."
+ "\n"+
                    "- ** PowerShell/BitsAdmin JA3 client match**" + "\n"+
                    "- **Client IP:** " + Flow.client.ipaddr + "\n"+
                    "- **Server IP:** " + Flow.server.ipaddr + "\n"+
                    "- **JA3 Client Value:** " + ja3 + "\n"+
                    "- **JA3 Client Match:** " + suspect_ja3_hashes[ja3],
                // Create the identity key by combining the server IP
                address, client IP address, and PowerShell JA3 hash
                identityKey: [
                    Flow.server.ipaddr,
                    Flow.client.ipaddr,
                    hash
                ].join('!!'),
                riskScore: 60,
            });
        }
    }
}

```

```
        identityTtl: 'hour'
      });
    }
  }
}
```