Configurer et valider des expressions régulières dans Cisco ESA et CES

Table des matières

Introduction

Ce document décrit comment ESA et CES utilisent des expressions régulières dans les filtres, les différences de comportement clés et la nécessité d'effectuer des tests avant l'application.

Informations générales

Ce document décrit comment Cisco Email Security Appliance (ESA) et Cisco Cloud Email Security (CES) gèrent les expressions régulières lorsqu'elles sont utilisées dans les filtres de messages et les filtres de contenu. Il s'attache tout particulièrement à comprendre le comportement des expressions régulières dans ces composants et leur interaction avec les en-têtes, le corps du texte et les pièces jointes des e-mails.

Il est important de préciser dès le début que le moteur d'expression régulière utilisé dans le module DLP se comporte différemment. Par conséquent, tout ce qui est décrit dans ce document s'applique exclusivement aux filtres de messages et aux filtres de contenu et ne s'applique pas aux stratégies DLP.

Lors de l'utilisation d'expressions régulières dans ESA, les administrateurs doivent comprendre que le contenu des e-mails n'est pas évalué de la même manière qu'il est visuellement affiché dans un client de messagerie. Les e-mails contiennent des informations d'enveloppe, des en-têtes structurés, des parties MIME et du contenu potentiellement codé. Par conséquent, les comparaisons effectuées par des filtres peuvent produire des résultats inattendus si la structure du message et le comportement d'expression régulière ne sont pas entièrement compris.

Pour cette raison, tout nouveau filtre qui utilise des expressions régulières peut toujours être activé en mode Surveillance avant l'application. Cela permet la validation par rapport au trafic réel et empêche le blocage ou l'impact sur les performances.

Dictionnaires et termes de recherche

Lors de la création d'un filtre de message ou de contenu, le terme entré dans de nombreuses conditions est interprété comme une expression régulière. Il s'agit d'un concept critique : même lorsque l'administrateur a l'intention de faire correspondre le texte littéral, ESA peut traiter l'entrée à l'aide de la logique regex.

Cela ne s'applique pas uniformément à tous les types de conditions. Par exemple, lors de la recherche d'une adresse IP spécifique dans certaines conditions structurées, la valeur n'est pas interprétée comme une expression régulière. Toutefois, lors d'une recherche dans l'en-tête Objet, le corps du message, un champ d'en-tête spécifique ou un nom de fichier joint, la valeur est généralement traitée comme un modèle d'expression régulière.

Un exemple courant illustre cela clairement. Supposons que l'objectif soit de bloquer les e-mails dont l'objet est :

```
Receipt number (123456)
```

Comme les parenthèses sont des caractères spéciaux dans les expressions régulières (utilisées pour le regroupement), elles doivent être échappées.

L'expression correcte serait :

```
Receipt number \(123456\)
```

Si les parenthèses ne sont pas échappées, le moteur d'expression régulière les interprète comme des opérateurs de regroupement plutôt que des caractères littéraux. Selon le modèle, cela peut provoquer des correspondances non souhaitées ou un comportement différent de celui attendu.

Pour cette raison, il est essentiel de comprendre quels caractères ont une signification particulière dans regex et de s'assurer qu'ils sont correctement échappés quand une correspondance littérale est requise.

Exemples de caractères spéciaux et de leur syntaxe d'échappement

La première colonne montre un exemple de texte contenant des caractères spéciaux, et la seconde montre comment la syntaxe correcte d'expression régulière doit être écrite pour correspondre à ce texte littéral dans Cisco ESA (Python-style regex).

Texte littéral à faire correspondre	Syntaxe d'expression régulière correcte
Numéro de réception (123456)	Numéro de réception \(123456\)
user@example.com	user@example\.com
www.test.abc	www.test\.abc
file_name.txt	file_name\.txt
Le prix est de 10,50	le prix est 10\.50
C:\Users\Admin	C:\\Users\\Admin
[CONFIDENTIEL]	\[CONFIDENTIEL\]
{facture}	\{facture\}
+34 600 123 456	\+34 600 123 456
question ?	question\ ?
Garantie à 100 %	Garantie à 100 % (% ne nécessite pas d'échappement)
astérisque *, symbole	symbole * astérisque
A|B	A|B
caret ^début	caret \^début
100 dollars	dollar \$100

Limitation de l'utilisation des expressions régulières

Les expressions régulières doivent être utilisées avec précaution et uniquement lorsque cela est nécessaire. Bien qu'elles offrent de puissantes fonctionnalités de correspondance, des expressions excessives ou mal conçues peuvent augmenter le temps de traitement des messages et produire des correspondances non souhaitées.

Une construction particulière qui nécessite de la prudence est .* , qui représente « n'importe quel caractère, zéro ou plusieurs fois ». Lorsqu'elle est placée au début ou à la fin d'une expression, elle peut entraîner un retour arrière excessif et une surcharge de traitement inutile.

La documentation Cisco indique que les entrées utilisant .* au début ou à la fin peuvent provoquer le verrouillage du système dans certaines conditions lors de la mise en correspondance de pièces MIME spécifiques. C'est pourquoi Cisco recommande d'éviter autant que possible d'utiliser l'approche avant ou arrière .*.

Dans de nombreux scénarios, les administrateurs utilisent des modèles tels que .*facture.*lorsqu'ils peuvent simplement écrire une facture et produire le même résultat pratique dans ESA. Comme le moteur d'analyse recherche déjà les zones de contenu appropriées, l'entourage d'un mot avec .* est souvent redondant et inefficace du point de vue informatique.

⚠ Mise en garde : La recommandation générale est de garder les expressions régulières aussi simples et précises que possible.

Filtres de messages, filtres de contenu et dictionnaires

Cisco ESA fournit plusieurs mécanismes pour évaluer les messages et appliquer des actions. Les filtres de messages fonctionnent au début du pipeline et utilisent une syntaxe de type script. Ils sont extrêmement flexibles et permettent une logique avancée impliquant des données d'enveloppe, des en-têtes et des propriétés de pièces jointes. Cependant, comme ils s'exécutent tôt dans la chaîne de traitement, des filtres de messages inefficaces peuvent nuire aux performances.

Les filtres de contenu sont configurés via l'interface graphique et fonctionnent une fois le message accepté. Pour la plupart des cas d'utilisation d'inspection de contenu, les filtres de contenu sont plus faciles à gérer et plus sûrs du point de vue des performances.

Dans les filtres de messages et les filtres de contenu, les expressions régulières peuvent être introduites soit directement dans une condition, soit indirectement par l'utilisation de dictionnaires.

Les dictionnaires permettent aux administrateurs de centraliser les termes de recherche réutilisables. Chaque entrée est écrite sur une ligne distincte et peut être du texte brut ou une expression régulière. Les dictionnaires prennent également en charge les caractères non ASCII, ce qui les rend adaptés aux environnements multilingues.

Dans certaines situations, certaines constructions d'expression régulière complexes ne peuvent pas se comporter de manière identique dans les dictionnaires. Dans ce cas, l'expression régulière doit être placée directement dans la condition de filtre plutôt qu'à l'intérieur du dictionnaire.

Cisco ESA permet de créer jusqu'à 150 dictionnaires de contenu. Par défaut, 100 dictionnaires peuvent être configurés, sauf si la limite est modifiée via l'interface de ligne de commande à l'aide de la commande dictionaryconfig.

Les dictionnaires peuvent également mettre en oeuvre une pondération de terme. Chaque terme peut se voir attribuer une pondération numérique et, lorsque l'ESA analyse un message, elle multiplie le nombre d'occurrences de ce terme par sa pondération. Le score résultant est comparé à un seuil défini dans le filtre. Ce modèle de notation permet une application plus souple et plus progressive des politiques.

En outre, les dictionnaires peuvent inclure des identificateurs intelligents, qui sont des détecteurs algorithmiques pour des modèles numériques structurés tels que des numéros de sécurité sociale ou des identificateurs bancaires.

Moteur d'expression régulière

Cisco ESA utilise des expressions régulières basées sur le style de module Python re. Bien que cela assure la compatibilité avec la syntaxe regex commune de Python, toutes les fonctionnalités avancées prises en charge dans les environnements Python complets ne sont pas nécessairement prises en charge dans ESA.

Pour une correspondance exacte des chaînes, les expressions doivent être ancrées à l'aide de ^ au début et de $ à la fin. Sans ces ancres, le moteur regex peut faire correspondre des sous-chaînes plutôt que des valeurs complètes.

Par exemple, l'expression :

```
sun.com
```

Correspondance des chaînes telles que :

```
thegodsunocommando
```

Cependant, l'expression :

```
^sun\.com$
```

Ne faites correspondre que la chaîne exacte sun.com.

Lors de la correspondance d'une chaîne vide, il est important de ne pas utiliser "", car cela correspond effectivement à toutes les chaînes. Au lieu de cela, l'expression correcte est :

```
^$
```

Puisque Cisco ESA utilise des expressions régulières de style Python, il existe deux façons d'effectuer une comparaison insensible à la casse.

Par défaut, comme indiqué, les expressions régulières sont sensibles à la casse. Cela implique de rechercher :

```
foo
```

Ne faites correspondre que foo, mais pas FOO, Foo ou fOo.

Si vous souhaitez effectuer une correspondance non sensible à la casse, vous pouvez utiliser l'indicateur en ligne (?i) au début de l'expression régulière. Ceci indique au moteur regex d'ignorer la casse pour le reste du motif.

Exemple :

```
(?i)foo
```

Cette expression correspond à :

- nourriture
- FOOT
- Poulet
- Ouf

Si vous voulez faire correspondre exactement la chaîne entière, en ignorant la casse, vous pouvez combiner l'indicateur insensible à la casse avec les ancres :

```
(?i)^foo$
```

Cela garantit que la valeur totale est exactement « alimentaire », quelle que soit la capitalisation.

Une autre solution (moins pratique) consisterait à définir explicitement toutes les combinaisons possibles à l'aide de classes de caractères, par exemple :

```
[Ff][Oo][Oo]
```

Cependant, cette approche devient difficile à maintenir et n'est pas recommandée lorsque l'indicateur (?i) peut être utilisé à la place.

Dans la plupart des scénarios ESA, la méthode préférée et la plus propre pour la correspondance insensible à la casse est d'utiliser :

```
(?i)
```

au début de l'expression régulière.

Caractères non ASCII et limites de mot

Dans les langues qui utilisent des jeux de caractères codés sur deux octets, les concepts de limites de mots ou de casse ne peuvent pas se comporter comme prévu. Les expressions complexes qui dépendent de constructions telles que \w peuvent produire des résultats incohérents lorsque le codage ou les paramètres régionaux sont inconnus.

Dans de tels cas, il peut être conseillé de désactiver l'application de limites de mots dans la configuration du dictionnaire ou de simplifier l'expression pour éviter la dépendance sur des classes de caractères ambiguës.

Lors de l'utilisation de dictionnaires non-ASCII, l'affichage CLI ne peut pas restituer correctement les caractères selon le codage du terminal. Dans ce cas, il est recommandé d'exporter le dictionnaire dans un fichier texte, de le modifier en externe et de le réimporter.

Écriture de filtres efficaces

L'efficacité est essentielle lors de l'écriture de filtres, en particulier dans les environnements à volume élevé. Une erreur courante est d'écrire de longues chaînes de conditions OR pour des correspondances similaires.

Par exemple, la vérification individuelle de dizaines d'extensions de pièces jointes force le moteur regex à s'initialiser de façon répétée. Cela augmente l'utilisation du processeur et réduit la maintenance.

Au lieu d'écrire plusieurs comparaisons séparées, les regrouper en utilisant l'alternance dans une seule expression régulière réduit considérablement la charge de traitement. Cela réduit le nombre

de fois où le moteur regex est appelé et rend le filtre plus facile à maintenir.

Une conception de filtre efficace ne concerne pas seulement la lisibilité : elle affecte directement les performances du système.

PDF et expressions régulières

La mise en correspondance du contenu dans des fichiers PDF peut produire des résultats inattendus selon la manière dont le PDF a été généré. Certains PDF ne contiennent pas d'espaces logiques ni de sauts de ligne dans leur représentation interne. Le moteur de balayage tente de reconstruire l'espacement logique en fonction du positionnement des mots.

Si un mot est construit à l'aide de plusieurs polices ou tailles de police, la représentation interne peut fragmenter le texte. Par exemple, le mot « callout » peut être interprété en interne comme « call out » ou « c a l lout ».

Dans ce cas, toute tentative de correspondance avec l'expression « callout » peut échouer car la représentation interne ne contient pas cette chaîne contiguë exacte. Les administrateurs doivent tenir compte de cette limitation lors de la conception de stratégies basées sur le contenu ciblant les pièces jointes PDF.

Test des expressions régulières

Tester des expressions régulières avant de les déployer en production est une exigence opérationnelle critique. Une expression régulière qui apparaît syntaxiquement correcte peut se comporter très différemment lorsqu'elle est évaluée par rapport au trafic de messagerie réel. En l'absence de tests appropriés, un filtre peut générer des faux positifs, ne pas détecter les modèles prévus, surcharger les performances ou perturber involontairement le flux d'e-mails légitimes.

Les essais doivent être envisagés comme un processus structuré en deux étapes afin de minimiser les risques avant d'activer un filtre en production.

Phase 1 - Conception et validation d'expressions régulières
La première phase porte sur la conception et la validation de l'expression régulière elle-même avant de l'intégrer dans Cisco ESA.

1. Utilisation de regex101 ou d'outils similaires

Les plates-formes en ligne telles que http://regex101.com (ou des outils équivalents) sont très utiles pendant la phase de conception. Lors de l'utilisation de ces outils, la saveur Python doit être sélectionnée pour s'approcher du moteur regex de l'ESA.

Ces plates-formes permettent aux administrateurs de :

- Valider la syntaxe correcte
- Vérifiez que les caractères spéciaux sont correctement protégés

- Tester les cas correspondants et non correspondants
- Visualiser le comportement du regroupement et du quantificateur
- Identifiez les constructions potentiellement gourmandes telles que .*

Cependant, ces outils simulent le comportement standard de Python regex et peuvent prendre en charge des fonctionnalités qui ne sont pas entièrement implémentées dans Cisco ESA. Par conséquent, ils doivent être considérés comme des outils de validation préliminaires plutôt que comme des tests de compatibilité définitifs.

2. Utilisation des modèles AI (ChatGPT, Copilot, ...)

Les assistants basés sur l'IA peuvent accélérer la création de regex, en particulier pour les scénarios de correspondance complexes. En décrivant le comportement souhaité en langage naturel, les administrateurs peuvent obtenir une proposition regex initiale qui peut ensuite être affinée.

Les outils IA sont particulièrement utiles pour :

- Génération d'expressions groupées complexes
- Conversion des exigences professionnelles en syntaxe regex
- Simplification des conditions longues basées sur OU en alternances groupées

Néanmoins, les expressions générées par l'IA doivent toujours être examinées de manière critique. Ils peuvent introduire des inefficacités, des constructions non prises en charge ou une logique trop complexe. L'aide à l'IA doit être traitée comme une aide à la rédaction et non comme une validation finale. Chaque expression générée par l'IA doit encore être testée à l'aide de méthodes de validation structurées.

Phase 2 - Validation du comportement des filtres dans Cisco ESA

Une fois l'expression elle-même validée, la deuxième phase se concentre sur la confirmation de son comportement au sein de Cisco ESA lorsqu'elle est appliquée au traitement réel des messages.

1. Utilisation de la fonction de trace dans la console CES

La fonctionnalité Trace de la console Cisco Email Security (CES) permet aux administrateurs de simuler et d'analyser le traitement d'un message spécifique. Il s'agit de l'une des méthodes les plus fiables pour valider le comportement des filtres avant leur application.

Trace offre une visibilité sur :

- Comment le message est analysé
- Quels filtres sont évalués ?

- Indique si la condition est déclenchée
- Ordre d'exécution des règles

Étant donné que ESA effectue l'analyse MIME, la normalisation d'en-tête et le décodage de contenu, le comportement à l'intérieur de l'appliance peut différer des outils de test regex externes. Pour obtenir des instructions détaillées, les administrateurs doivent consulter la documentation officielle de Cisco :

https://www.cisco.com/c/en/us/td/docs/security/ces/ces_16-0-3/user_guide/b_ESA_Admin_Guide_ces_16-0-3/b_ESA_Admin_Guide_12_1_chapter_0101001.html

L'utilisation de Trace garantit que le filtre se comporte comme prévu dans le moteur de traitement réel.

2. Création du filtre avec une action de consignation

Une autre approche sûre et recommandée consiste à déployer le filtre avec une action non perturbatrice, telle que la journalisation, au lieu d'appliquer une action agressive telle que l'abandon, le renvoi ou la mise en quarantaine des messages.

En configurant le filtre pour consigner une entrée en cas de correspondance, les administrateurs peuvent :

- Observer la fréquence de correspondance
- Détecter les déclencheurs inattendus
- Valider l'impact sur les performances
- Analyser le comportement réel du trafic

Cette approche place efficacement le filtre dans une phase de surveillance contrôlée au sein du trafic de production. Une fois que la validation suffisante est terminée et que le comportement est confirmé, l'action peut passer en mode d'application en toute sécurité.

Présentation de l'expression dans un filtre de contenu et dans un dictionnaire

Une fois que l'expression régulière a été correctement conçue et validée, l'étape suivante consiste à comprendre comment elle doit être entrée dans Cisco ESA. La syntaxe peut être légèrement différente selon que l'expression est configurée directement dans une condition de filtre de contenu ou dans un dictionnaire. Cette différence est souvent source de confusion.

Présentation de l'expression dans un filtre de contenu

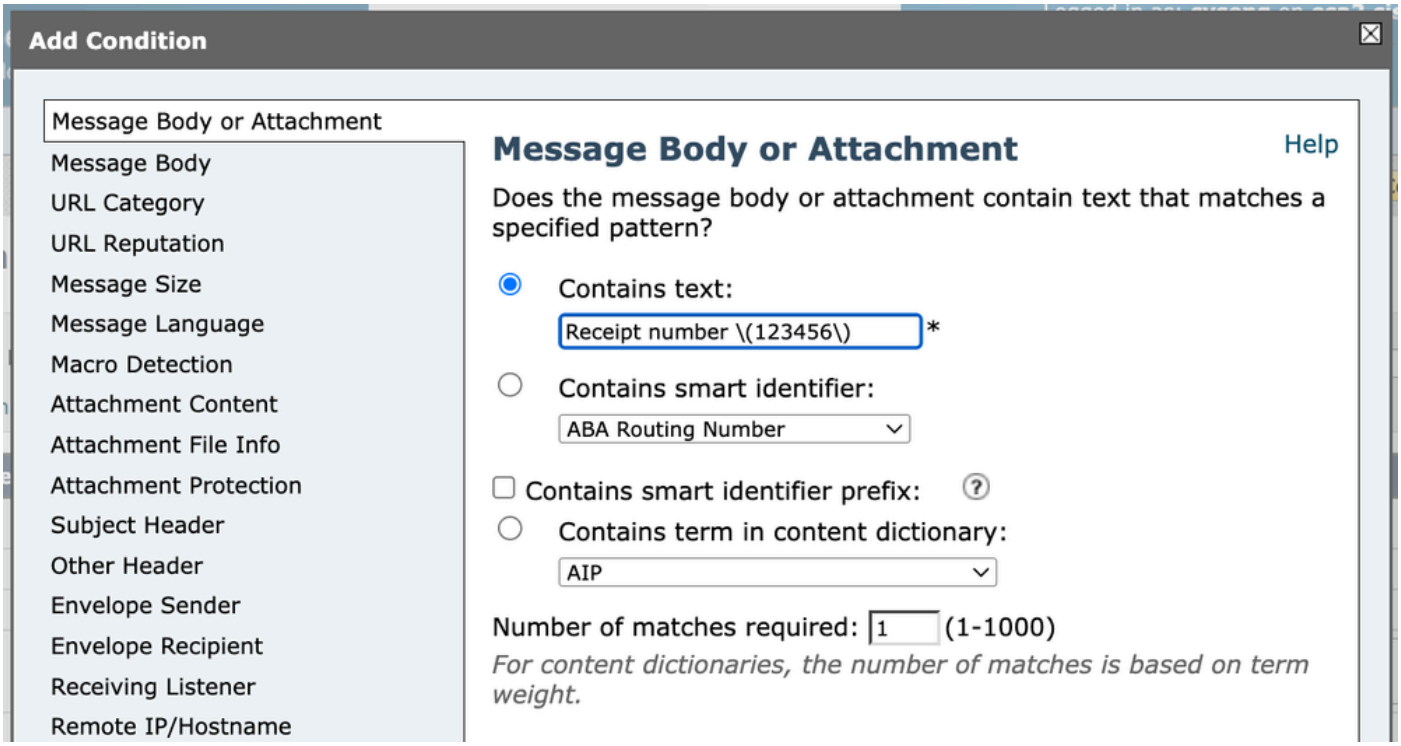
Lors de la configuration d'une condition de filtre de contenu (par exemple, la correspondance avec l'en-tête Subject), l'expression régulière doit être entrée dans le champ de condition. Si nous voulons faire correspondre le texte littéral :

```
Receipt number (123456)
```

Nous devons échapper les parenthèses car ce sont des caractères spéciaux dans les expressions régulières.

Par conséquent, le regex lui-même doit être écrit comme suit :

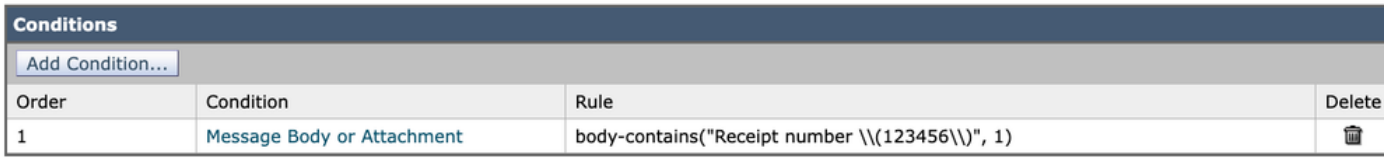
```
Receipt number \(123456\)
```



Filtre de contenu 1

Cependant, lorsque vous affichez la condition de filtre complet dans l'interface graphique utilisateur ou le résultat de la configuration avancée, elle peut apparaître comme suit :

```
subject == "Receipt number \\(123456\\)"
```



Filtre de contenu 2

Cela peut être déroutant à première vue. La raison de la double barre oblique inverse (\\) est que la barre oblique inverse elle-même est également un caractère spécial dans les chaînes entre guillemets. Dans ce contexte, une barre oblique inverse est utilisée pour échapper à la parenthèse du moteur d'expression régulière, et la deuxième barre oblique inverse est utilisée pour échapper à la barre oblique inverse dans la chaîne entre guillemets.

En termes pratiques :

\(123456\) est l'expression régulière réelle.

\\(représente le système \(dans une chaîne de configuration entre guillemets.

Bien qu'elle apparaisse différente lorsqu'elle est affichée, l'expression régulière logique en cours d'évaluation reste :

Numéro de réception \(123456\)

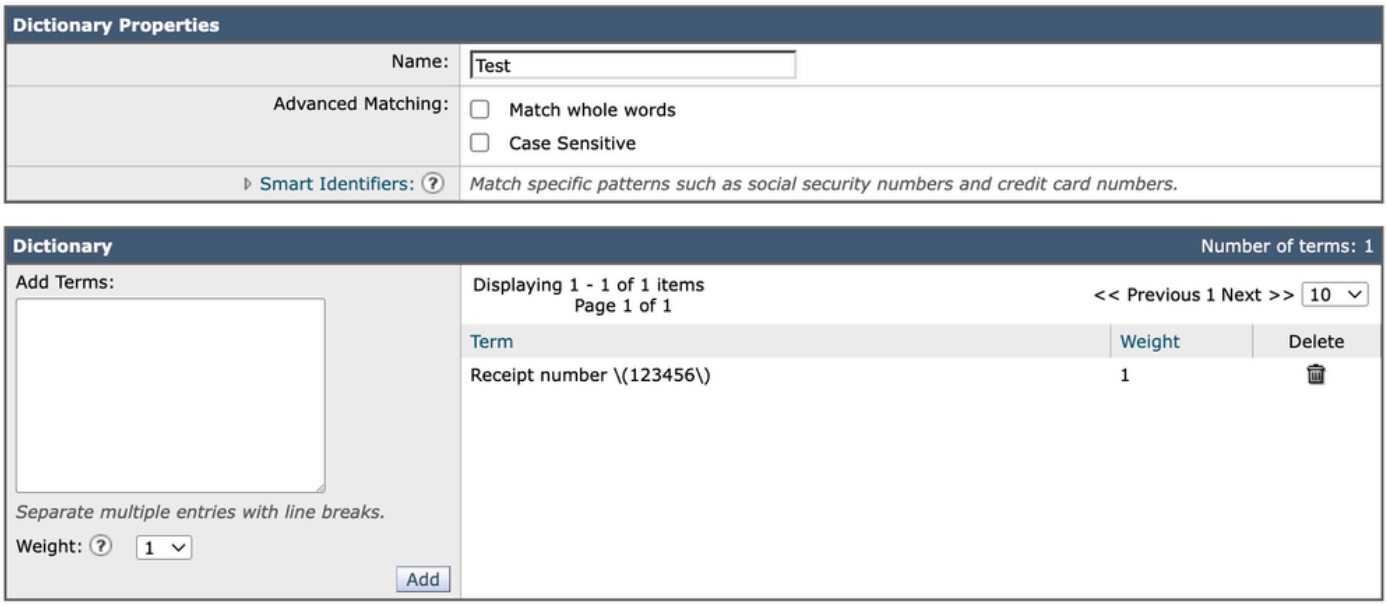
Il s'agit simplement d'une chaîne qui s'échappe dans la sortie de configuration.

Présentation de l'expression dans un dictionnaire

Lors de l'ajout de la même expression à un dictionnaire, l'entrée est introduite directement comme suit :

```
Receipt number \(123456\)
```

Dans ce cas, il continue à s'afficher exactement tel qu'il est écrit. Contrairement à la représentation graphique du filtre de contenu, les dictionnaires ne nécessitent pas de couches d'échappement supplémentaires dans leur format de configuration visuelle.



Dictionnaire

Chaque entrée de dictionnaire est évaluée en tant que texte brut ou expression régulière selon sa structure. Si des caractères spéciaux sont inclus (comme des parenthèses dans ce cas), l'expression doit déjà être correctement échappée lors de sa saisie.

À propos de « Correspondance de mots entiers »

Lors de la configuration d'un dictionnaire, il existe une option appelée « Correspondance des mots entiers ». Dans de nombreux cas, il est recommandé de ne pas utiliser ce paramètre lors de l'utilisation d'expressions régulières.

La raison en est que le comportement de limite de mot peut être contrôlé plus précisément à l'aide d'ancres d'expression régulière.

Exemple :

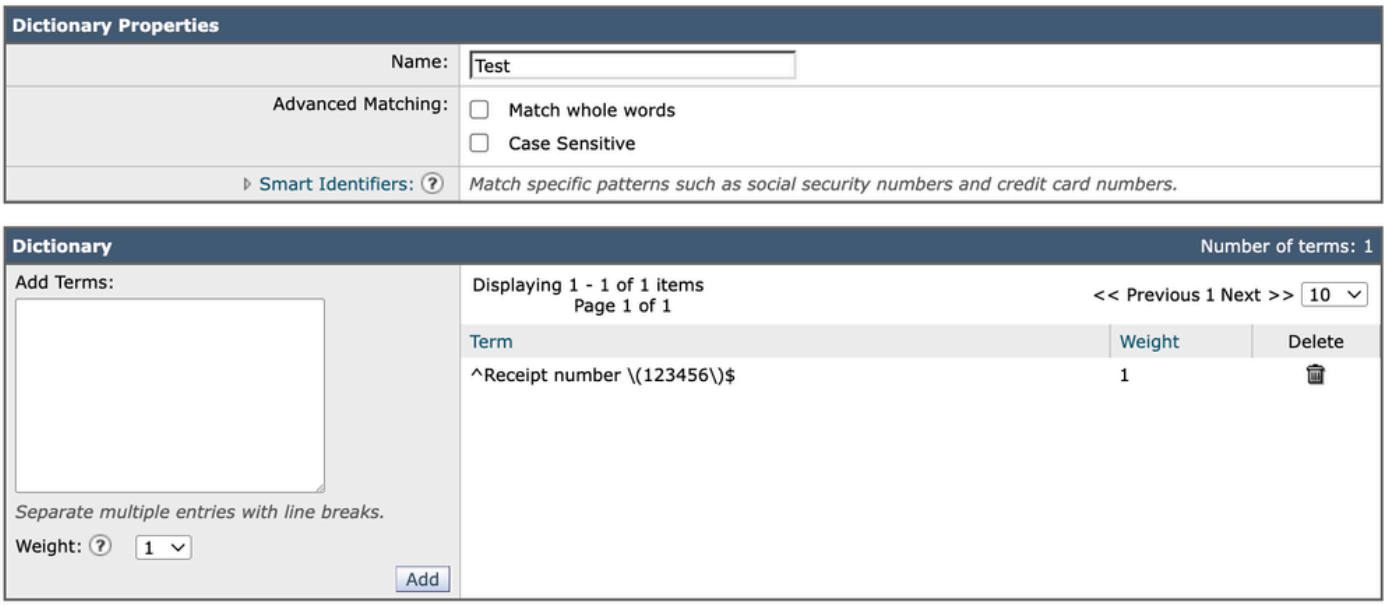
^ garantit que la correspondance commence au début.

$ s'assure que la correspondance se termine à la fin.

Utilisation d'ancres telles que :

```
^Receipt number \(123456\)$
```

Fournit un contrôle explicite et prévisible sur le comportement de correspondance exact. Cette approche évite toute ambiguïté potentielle liée à l'interprétation des limites des mots, en particulier dans les environnements multilingues ou non ASCII.



Dictionnaire 2

Pour cette raison, il est généralement préférable de gérer la précision de la correspondance directement dans l'expression régulière plutôt que de s'appuyer sur l'option « Match Whole Words ».

La compréhension de ces différences subtiles entre les filtres de contenu et les dictionnaires garantit que les expressions se comportent de manière cohérente et réduit le risque d'erreurs de configuration lors de la mise en oeuvre.

Classement des coûts Regex dans Cisco ESA

Lors de l'utilisation d'expressions régulières dans Cisco ESA, l'impact sur les performances dépend en grande partie de la quantité de texte que le moteur doit analyser et de la quantité de retour arrière qu'il doit effectuer. Étant donné que ESA doit évaluer des corps de message entiers, des parties MIME et même des pièces jointes décodées, des modèles inefficaces peuvent augmenter considérablement l'utilisation du processeur.

Il s'agit d'un classement pratique allant du coût de calcul le plus élevé au plus faible.

Les plus onéreux : modèles à haut risque

Ces expressions peuvent avoir un impact considérable sur les performances, en particulier sur les messages volumineux.

Quantificateurs imbriqués (cas le plus défavorable)

Exemples:

```
(.*)+
(.+)+
```

```
(\S+)+
```

Ils sont extrêmement dangereux car ils créent des scénarios de retour en arrière exponentiels.

Un quantificateur à l'intérieur d'un autre quantificateur force le moteur regex à essayer plusieurs combinaisons avant d'échouer.

Dans le trafic réel, cela peut provoquer de sérieux pics de CPU.

Recommandation : Évitez les quantificateurs imbriqués non limités et ambigus.

Gourmand .* Suivi d'un modèle requis

Exemple :

```
.*text
.*\/\?text
```

Ce modèle consomme d'abord l'intégralité du message, puis effectue un retour arrière caractère par caractère jusqu'à ce qu'il trouve la sous-chaîne requise.

Si le modèle n'est pas présent — ou apparaît près de la fin — le moteur fait marche arrière et teste le jeton requis à plusieurs positions, ce qui augmente le coût du processeur.

Dans l'ESA, où les corps peuvent être volumineux et inclure du contenu MIME, cela devient très rapidement coûteux.

Recommandation : ne pas ajouter .* pour détecter les sous-chaînes. ESA effectue déjà des recherches dans le contenu évalué, et les caractères génériques de tête ne font qu'augmenter le retour arrière et l'utilisation du CPU.

```
text$
\/\?text$
```

Grandes alternatives avec préfixes partagés

Exemple :

```
(a.*b|a.*c|a.*d)
```

Lorsque plusieurs alternatives partagent la même structure, le moteur évalue chaque branche de manière séquentielle.

Si les premières branches correspondent presque mais échouent tard, le moteur effectue de nombreuses nouvelles tentatives.

Cela augmente considérablement le temps d'évaluation.

Coût Moyen — À Utiliser Avec Précaution

Ces modèles ne sont pas catastrophiques, mais ils peuvent être inefficaces.

Large .* Utilisation

Exemple :

```
https://.*\?text
```

Bien qu'il ne soit pas exponentiel, .* permet toujours une correspondance illimitée. Si la sous-chaîne attendue n'apparaît pas rapidement, le moteur analyse de grandes parties du message.

Dans ESA, c'est fréquent lors de l'analyse des corps d'e-mail pour les URL d'hameçonnage.

Quantificateurs paresseux (+?, *?)

Exemple :

```
\S+?
.*?
```

Les quantificateurs paresseux modifient la stratégie de correspondance (shortest-first). Ils peuvent réduire la surcorrespondance dans certains modèles, mais dans les charges de travail de recherche volumineuses, ils peuvent augmenter le nombre de tentatives lorsque le jeton de terminaison est en retard ou manquant.

Dans de nombreux cas d'utilisation ESA, elles ne présentent pas d'avantages réels et peuvent entraîner des tentatives internes inutiles.

Classes de caractères très génériques

Exemples:

```
\S+
.+
```

Ils permettent une large plage de correspondances, augmentant le nombre de chemins de retour en arrière potentiels.

Des classes de caractères plus spécifiques sont toujours préférables.

Faible coût — Modèles sûrs et efficaces

Ils sont recommandés pour les environnements ESA de production.

Littéraux fixes

Exemples:

```
text
iw\.adc
```

Les chaînes littérales sont les correspondances les plus efficaces possibles. Le moteur effectue des comparaisons directes avec une surcharge minimale.

Utiliser des ancres pour limiter la portée de la recherche

Lorsque la correspondance est attendue à une position spécifique, pensez à ancrer le motif à l'aide de ^ ou de $. Les ancres limitent l'évaluation à des positions fixes et empêchent le moteur d'analyser inutilement l'ensemble du contenu. Cela peut réduire le retour arrière et améliorer les performances, en particulier dans les corps de message volumineux ou les en-têtes structurés.

```
^Invoice$
```

Classes de caractères spécifiques

```
[A-Za-z0-9.-]+
[^/\s]+
```

Elles restreignent ce qui peut correspondre, réduisant considérablement l'espace de recherche et limitant le retour arrière.

Modèles structurés et sous contraintes

Exemple :

```
https?:\/\/[A-Za-z0-9.-]+(?:\/[^?\s]*)*\/\?text
```

- Le domaine est fixe.
- Ne pas utiliser .*.
- ne contient pas de modèles imbriqués catastrophiques (par exemple, (.*)+)
- Pas d'opérateurs paresseux inutiles.
- Chaque section est contrainte.

Cela réduit considérablement l'impact sur le processeur par rapport à une correspondance générique étendue.

Conseils pratiques pour Cisco ESA

Lors de la conception de regex pour les filtres de messages ou de contenu :

1. Plus le modèle est spécifique, meilleures sont les performances.
2. Évitez .* sauf si c'est vraiment nécessaire — et surtout évitez de placer les jetons requis après lui.
3. N'utilisez jamais de quantificateurs imbriqués.
4. Préférez les classes de caractères explicites aux caractères génériques.
5. Testez toujours les nouvelles expressions en mode Surveillance avant de les appliquer.

Comparaison des performances Regex (contexte Cisco ESA)

Modèle	Recommandé	Risque De Retour En Arrière	Impact ESA	Alternative recommandée
https?:\/\/.*/\?text.*	Non	Élevé	Supérieur	^https?:\/\/[A-Za-z0-9.-]+(?:\/[^?\s]*)*\/\?text
https?:\/\/.*\?text	⚠ Avec prudence	Moyenne à élevée	Moyenne à élevée	^https?:\/\/[^?\s]+\?text$
https?:\/\/.*	Non	Moyenne à élevée	Moyen	^https?:\/\/[A-Za-z0-9.-]+(?:\/[^\s]*)*

.*mot de passe	Non	Élevé	Supérieur	password$
.*texte.*	Non	Élevé	Supérieur	texte
.*(facture|paiement|virement)	Non	Élevé	Supérieur	(facture|paiement|transfert)$
(.+)+	Jamais	Très élevé (exponentiel)	Severe (grave)	Restructuration sans quantificateurs imbriqués (exemple .+)
.*@.*	Non	Élevé	Supérieur	[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}
\S+?	Pas idéal	Moyen	Moyen	\S+ ou une classe plus spécifique comme [A-Za-z0-9.-]+
.*\/admin	Non	Élevé	Supérieur	\/admin$
.*(connexion|vérification).*	Non	Élevé	Supérieur	(connexion|vérification)
^.*texte	Non	Élevé	Supérieur	text$ (ou ^text si la position compte)

Conclusion

Les expressions régulières constituent un outil puissant et flexible au sein de Cisco ESA, permettant une inspection précise du contenu et une application avancée des politiques dans les filtres de messages et les filtres de contenu. Toutefois, cette flexibilité s'accompagne de responsabilités. Des expressions mal conçues ou insuffisamment testées peuvent entraîner des faux positifs, des détections manquées, une dégradation des performances ou une interruption involontaire du trafic de messagerie légitime.

Pour cette raison, l'utilisation d'expressions régulières dans l'ESA doit toujours être structurée et disciplinée. La phase de création doit garantir que l'expression est syntaxiquement correcte, correctement échappée, efficace et logiquement alignée avec l'objectif visé. Les outils externes et la génération assistée par IA peuvent considérablement accélérer ce processus, mais ils ne doivent jamais remplacer une validation minutieuse.

Tout aussi importante est la phase de validation au sein de l'environnement ESA lui-même. Étant

donné que l'ESA traite les messages via l'analyse MIME, la normalisation d'en-tête et le décodage de contenu, le comportement réel peut différer des attentes théoriques. L'utilisation d'outils tels que Trace et le déploiement initial de filtres en mode de journalisation ou de surveillance permet aux administrateurs de confirmer le comportement correct sans risque opérationnel.

En résumé, les expressions régulières doivent être aussi simples que possible, testées minutieusement et déployées avec prudence. Un filtre bien conçu et correctement validé permet non seulement d'appliquer efficacement les politiques, mais également de protéger la stabilité du système et de garantir un comportement prévisible dans les environnements de production.

Documentation

Pour obtenir des détails techniques supplémentaires et des conseils officiels sur la manière dont les expressions régulières sont implémentées et utilisées dans Cisco ESA, les administrateurs doivent consulter la documentation produit Cisco

La section « Expressions régulières dans les règles » fournit une vue d'ensemble de la façon dont les expressions régulières sont évaluées dans les filtres de messages et les filtres de contenu, y compris les considérations de syntaxe et l'utilisation dans les conditions de règle.

https://www.cisco.com/c/en/us/td/docs/security/ces/ces_16-0-3/user_guide/b_ESA_Admin_Guide_ces_16-0-3/b_ESA_Admin_Guide_12_1_chapter_01000.html#con_1192755

La section « Instructions pour l'utilisation des expressions régulières » propose des recommandations pratiques sur la syntaxe correcte, l'ancrage des expressions, la gestion des caractères spéciaux et l'élimination des erreurs courantes susceptibles d'affecter les performances ou la précision de la correspondance.

https://www.cisco.com/c/en/us/td/docs/security/ces/ces_16-0-3/user_guide/b_ESA_Admin_Guide_ces_16-0-3/b_ESA_Admin_Guide_12_1_chapter_01000.html#con_1125696

Il est vivement recommandé de consulter ces ressources officielles lors de la conception ou du dépannage de filtres qui reposent sur des expressions régulières, car ils fournissent des conseils faisant autorité et alignés sur la version spécifique d'AsyncOS utilisée.

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.