Configurazione e convalida delle espressioni regolari in Cisco ESA e CES

Sommario

Introduzione

Questo documento descrive come ESA e CES utilizzano espressioni regolari nei filtri, le differenze di comportamento chiave e la necessità di eseguire test prima dell'applicazione.

Premesse

In questo documento viene descritto come Cisco Email Security Appliance (ESA) e Cisco Cloud Email Security (CES) gestiscono le espressioni regolari quando vengono usate nei filtri messaggi e contenuti. Si occupa in particolare di comprendere come si comportano le espressioni regolari in questi componenti e come interagiscono con le intestazioni e-mail, il contenuto del corpo e gli allegati.

È importante chiarire fin dall'inizio che il motore delle espressioni regolari utilizzato nel modulo DLP funziona in modo diverso. Pertanto, tutte le informazioni descritte in questo documento si applicano esclusivamente ai filtri messaggi e ai filtri contenuti e non ai criteri di prevenzione della perdita dei dati.

Quando si utilizzano espressioni regolari in ESA, gli amministratori devono comprendere che il contenuto di posta elettronica non viene valutato allo stesso modo in cui viene visualizzato visivamente in un client di posta. I messaggi di posta elettronica contengono informazioni sulla busta, intestazioni strutturate, parti MIME e contenuto potenzialmente codificato. Di conseguenza, i confronti eseguiti dai filtri possono produrre risultati imprevisti se la struttura dei messaggi e il comportamento di regex non sono pienamente compresi.

Per questo motivo, qualsiasi nuovo filtro che utilizza espressioni regolari può sempre essere attivato in modalità di monitoraggio prima dell'applicazione. In questo modo è possibile eseguire la convalida rispetto al traffico reale ed evitare blocchi involontari o impatti sulle prestazioni.

Dizionari e termini di ricerca

Quando si crea un filtro messaggi o un filtro contenuti, il termine immesso in molte condizioni viene interpretato come espressione regolare. Si tratta di un concetto critico: anche quando l'amministratore intende far corrispondere il testo letterale, ESA può elaborare l'input utilizzando la logica regex.

Ciò non si applica in modo uniforme a tutti i tipi di condizione. Ad esempio, quando si cerca un indirizzo IP specifico in determinate condizioni strutturate, il valore non viene interpretato come espressione regolare. Tuttavia, quando si esegue la ricerca all'interno dell'intestazione Subject, del corpo del messaggio, di un campo di intestazione specifico o di un nome file allegato, il valore viene in genere considerato come un modello regex.

Un esempio comune lo illustra chiaramente. Si supponga che l'obiettivo sia bloccare le e-mail con l'oggetto:

```
Receipt number (123456)
```

Poiché le parentesi sono caratteri speciali nelle espressioni regolari (utilizzate per il raggruppamento), è necessario che sia applicato un carattere di escape.

L'espressione corretta sarebbe:

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

Se le parentesi non presentano un carattere di escape, il modulo di gestione regex le interpreta come operatori di raggruppamento anziché come caratteri letterali. A seconda del modello, ciò può causare corrispondenze non intenzionali o comportamenti diversi da quelli previsti.

Per questo motivo, è essenziale capire quali caratteri hanno un significato speciale in regex e assicurarsi che siano correttamente evasi quando è richiesta la corrispondenza letterale.

Esempi di caratteri speciali e relativa sintassi con escape

La prima colonna mostra un testo di esempio contenente caratteri speciali e la seconda colonna mostra come la sintassi corretta dell'espressione regolare deve essere scritta per corrispondere al testo letterale in Cisco ESA (Python-style regex).

Testo letterale da abbinare	Correggi sintassi espressione regolare
Numero ricevuta (123456)	Numero ricevuta \(123456\)
user@example.com	user@example\.com
www.test.ab c	www\.test\.abc
file_name.txt	file_name\.txt
il prezzo è 10,50	il prezzo è 10\.50
C:\Users\Admin	C:\\Users\\Admin
[RISERVATO]	\[RISERVATO\]
{fattura}	\{fattura\}
+34 600 123 456	\+34 600 123 456
domanda?	domanda\?
100% garantito	100% garantito (% non richiede escape)
asterisco *	asterisco *
A|B	A|B
accento circonflesso ^start	accento circonflesso \^inizio
$ 100	dollaro \$100

Limitazione dell'utilizzo delle espressioni regolari

Le espressioni regolari devono essere utilizzate con attenzione e solo quando necessario. Sebbene offrano potenti funzionalità di corrispondenza, espressioni eccessive o mal progettate possono aumentare il tempo di elaborazione dei messaggi e produrre corrispondenze non intenzionali.

Un particolare costrutto che richiede attenzione è .* , che rappresenta "qualsiasi carattere, zero o più volte". Quando viene posizionato all'inizio o alla fine di un'espressione, può causare un eccessivo backtracking e un inutile sovraccarico di elaborazione.

La documentazione Cisco indica che le voci che utilizzano .* all'inizio o alla fine possono causare il blocco del sistema in determinate condizioni quando si corrispondono parti MIME specifiche. Per questo motivo, Cisco consiglia di evitare l'uso di .* iniziali o finali quando possibile.

In molti scenari, gli amministratori utilizzano modelli come .*fattura.*quando potrebbero semplicemente scrivere la fattura e produrre lo stesso risultato pratico in ESA. Poiché il motore di scansione esegue già la ricerca nelle aree di contenuto rilevanti, circondare una parola con .* è spesso ridondante e inefficiente dal punto di vista computazionale.

⚠ Attenzione: La raccomandazione generale è quella di mantenere le espressioni regolari il più possibile semplici e precise.

Filtri messaggi, filtri contenuti e dizionari

Cisco ESA fornisce diversi meccanismi per valutare i messaggi e applicare le azioni. I filtri messaggi funzionano all'inizio della pipeline e utilizzano una sintassi in stile script. Sono estremamente flessibili e consentono una logica avanzata che coinvolge dati di busta, intestazioni e proprietà degli allegati. Tuttavia, poiché vengono eseguiti all'inizio della catena di elaborazione, l'inefficienza dei filtri messaggi può avere un impatto negativo sulle prestazioni.

I filtri dei contenuti vengono configurati tramite l'interfaccia grafica e funzionano dopo l'accettazione del messaggio. Nella maggior parte dei casi di utilizzo relativi all'ispezione dei contenuti, i filtri dei contenuti sono più facili da gestire e più sicuri dal punto di vista delle prestazioni.

Sia nei filtri messaggi che nei filtri contenuti, le espressioni regolari possono essere introdotte direttamente in una condizione o indirettamente tramite l'utilizzo di dizionari.

I dizionari consentono agli amministratori di centralizzare i termini di ricerca riutilizzabili. Ogni voce viene scritta su una riga separata e può essere costituita da testo normale o da un'espressione regolare. I dizionari supportano anche caratteri non ASCII, rendendoli adatti ad ambienti multilingue.

In alcune situazioni, alcuni costrutti di espressioni regolari complesse non possono comportarsi in modo identico all'interno dei dizionari. In questo caso, l'espressione regolare deve essere inserita direttamente nella condizione del filtro anziché all'interno del dizionario.

Cisco ESA consente di creare fino a 150 dizionari di contenuti. Per impostazione predefinita, è possibile configurare 100 dizionari a meno che il limite non venga modificato tramite la CLI con il comando dictionaryconfig.

I dizionari possono anche implementare la ponderazione dei termini. A ogni termine può essere assegnato un peso numerico e, quando l'ESA analizza un messaggio, moltiplica il numero di occorrenze di quel termine per il suo peso. Il punteggio risultante viene confrontato con una soglia definita nel filtro. Questo modello di punteggio consente un'applicazione più flessibile e graduale delle politiche.

Inoltre, i dizionari possono includere identificatori intelligenti, ovvero rilevatori algoritmici per modelli numerici strutturati quali numeri di previdenza sociale o identificatori bancari.

Motore espressione regolare

Cisco ESA utilizza espressioni regolari basate sullo stile di modulo Python re. Anche se questo fornisce la compatibilità con la sintassi regex Python comune, non tutte le funzionalità avanzate supportate in ambienti Python completi sono necessariamente supportate in ESA.

Per ottenere una corrispondenza esatta tra stringhe, le espressioni devono essere ancorate utilizzando ^ all'inizio e $ alla fine. Senza questi ancoraggi, il motore regex può corrispondere a sottostringhe anziché a valori completi.

Ad esempio, l'espressione:

```
sun.com
```

Stringhe di corrispondenza quali:

```
thegodsunocommando
```

Tuttavia, l'espressione

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

Trovare solo la stringa esatta sun.com.

Quando si trova una stringa vuota, è importante non utilizzare "", in quanto in questo modo vengono trovate tutte le stringhe. Al contrario, l'espressione corretta è:

```
^$
```

Poiché Cisco ESA utilizza espressioni regolari in stile Python, esistono un paio di modi per eseguire un confronto senza distinzione tra maiuscole e minuscole.

Per impostazione predefinita, come indicato, le espressioni regolari distinguono tra maiuscole e minuscole. Ciò significa cercare:

```
foo
```

Corrispondono solo a foo, ma non a FOO, Foo o fOo.

Se si desidera eseguire una corrispondenza senza distinzione tra maiuscole e minuscole, è possibile utilizzare il contrassegno in linea (?i) all'inizio dell'espressione regolare. Ciò indica al motore regex di ignorare la distinzione tra maiuscole e minuscole per il resto del modello.

Ad esempio:

```
(?i)foo
```

Questa espressione corrisponde:

- cibo
- FOO
- Cibo
- Oo

Se si desidera ottenere una corrispondenza esatta con l'intera stringa, ignorando la distinzione tra maiuscole e minuscole, è possibile combinare il contrassegno senza distinzione tra maiuscole e minuscole con gli ancoraggi:

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

Questo assicura che il valore completo sia esattamente "foo", indipendentemente dalla capitalizzazione.

Un'altra alternativa (meno pratica) sarebbe quella di definire esplicitamente tutte le possibili combinazioni utilizzando le classi di caratteri, ad esempio:

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

Tuttavia, questo approccio diventa difficile da mantenere e non è consigliato quando è possibile utilizzare il flag (?i).

Nella maggior parte degli scenari ESA, il metodo preferibile e più chiaro per la corrispondenza senza distinzione tra maiuscole e minuscole è utilizzare:

```
(?i)
```

all'inizio dell'espressione regolare.

Caratteri non ASCII e limiti di parola

Nei linguaggi che utilizzano set di caratteri a byte doppio, il comportamento dei concetti di delimitazione delle parole o maiuscole/minuscole non è quello previsto. Le espressioni complesse che dipendono da costrutti quali \w possono produrre risultati incoerenti quando la codifica o le impostazioni internazionali sono sconosciute.

In questi casi, può essere consigliabile disabilitare l'imposizione di limiti di parola nella configurazione del dizionario o semplificare l'espressione per evitare la dipendenza da classi di caratteri ambigui.

Quando si utilizzano dizionari non ASCII, la visualizzazione CLI non è in grado di eseguire correttamente il rendering dei caratteri a seconda della codifica del terminale. In questi casi, si consiglia di esportare il dizionario in un file di testo, modificarlo esternamente e reimportarlo.

Scrittura di filtri efficienti

L'efficienza è fondamentale per la scrittura di filtri, soprattutto in ambienti con volumi elevati. Un errore comune è scrivere lunghe catene di condizioni OR per corrispondenze simili.

Ad esempio, se si selezionano decine di estensioni di attacco singolarmente, il motore regex viene inizializzato ripetutamente. Ciò aumenta l'utilizzo della CPU e riduce la manutenibilità.

Anziché scrivere molti confronti distinti, raggruppandoli mediante alternanza in una singola espressione regolare riduce in modo significativo il sovraccarico di elaborazione. In questo modo si riduce il numero di volte in cui il motore regex viene richiamato e si semplifica la gestione del filtro.

L'efficienza del filtro non riguarda solo la leggibilità, ma influisce direttamente sulle prestazioni del sistema.

PDF ed espressioni regolari

La corrispondenza del contenuto all'interno dei file PDF può produrre risultati imprevisti a seconda di come è stato generato il PDF. Alcuni PDF non contengono spazi logici o interruzioni di riga nella rappresentazione interna. Il motore di digitalizzazione tenta di ricostruire la spaziatura logica in base al posizionamento delle parole.

Se una parola è costruita utilizzando più tipi di carattere o dimensioni di carattere, la rappresentazione interna può frammentare il testo. Ad esempio, la parola "callout" può essere interpretata internamente come "callout" o "c a l lout".

In questi casi, il tentativo di trovare una corrispondenza con l'espressione "callout" può avere esito negativo perché la rappresentazione interna non contiene la stringa contigua esatta. Gli amministratori devono essere consapevoli di questo limite quando progettano policy basate sul contenuto destinate agli allegati PDF.

Test delle espressioni regolari

Il test delle espressioni regolari prima della loro distribuzione in produzione è un requisito operativo critico. Un'espressione regolare che appare sintatticamente corretta può comportarsi in modo molto diverso se valutata rispetto al traffico e-mail reale. Senza test appropriati, un filtro può generare falsi positivi, non rilevare i modelli desiderati, introdurre un sovraccarico delle prestazioni o interrompere in modo non intenzionale il flusso di posta legittimo.

I test devono essere affrontati come un processo strutturato in due fasi per ridurre al minimo i rischi prima di attivare un filtro in produzione.

Fase 1 - Progettazione e convalida delle espressioni regolari
La prima fase si concentra sulla progettazione e convalida dell'espressione regolare stessa prima di integrarla in Cisco ESA.

1. Uso di regex101 o di strumenti simili

Le piattaforme online come http://regex101.com (o strumenti equivalenti) sono molto utili durante la fase di progettazione. Quando si utilizzano questi strumenti, il sapore Python deve essere selezionato per approssimare il motore regex dell'ESA.

Queste piattaforme consentono agli amministratori di:

- Convalida la correttezza della sintassi
- Confermare che i caratteri speciali siano correttamente sottoposti a escape
- Test dei casi corrispondenti e non corrispondenti

- Visualizza comportamento raggruppamento e quantificatore
- Identificare i costrutti potenzialmente greedy, ad esempio .*

Tuttavia, questi strumenti simulano il comportamento standard dei Python regex e possono supportare funzionalità non completamente implementate in Cisco ESA. Pertanto, devono essere considerati strumenti di convalida preliminari piuttosto che test di compatibilità definitivi.

2. Uso di modelli AI (ChatGPT, Copilot, ...)

Gli assistenti basati su AI possono accelerare la creazione di regex, in particolare per scenari di corrispondenza complessi. Descrivendo il comportamento desiderato in linguaggio naturale, gli amministratori possono ottenere una proposta regex iniziale che può quindi essere perfezionata.

Gli strumenti AI sono particolarmente utili per:

- Generazione di espressioni di raggruppamento complesse
- Conversione dei requisiti aziendali in sintassi regex
- Semplificazione di condizioni lunghe basate su OR in alternanze raggruppate

Tuttavia, le espressioni generate dall'IA devono sempre essere riviste in modo critico. Possono introdurre inefficienze, costrutti non supportati o logica eccessivamente complessa. L'assistenza all'IA deve essere trattata come un aiuto redazionale e non come una convalida finale. Ogni espressione generata dall'IA deve comunque essere verificata utilizzando metodi di convalida strutturati.

Fase 2 - Convalida del comportamento del filtro in Cisco ESA

Una volta che l'espressione stessa è stata convalidata, la seconda fase si concentra sulla conferma del suo comportamento all'interno di Cisco ESA quando applicata all'elaborazione reale dei messaggi.

1. Utilizzo della funzione di traccia nella console CES

La funzione Trace nella console Cisco Email Security (CES) consente agli amministratori di simulare e analizzare la modalità di elaborazione di un messaggio specifico. Questo è uno dei metodi più affidabili per convalidare il comportamento del filtro prima dell'applicazione.

Trace offre visibilità su:

- Modalità di analisi del messaggio
- Quali filtri vengono valutati
- Indica se la condizione è attivata
- Ordine di esecuzione delle regole

Poiché ESA esegue l'analisi MIME, la normalizzazione dell'intestazione e la decodifica del contenuto, il comportamento all'interno dell'accessorio può essere diverso dagli strumenti di test regex esterni. Per istruzioni dettagliate, gli amministratori devono consultare la documentazione ufficiale di 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

Trace garantisce che il filtro si comporti come previsto all'interno del motore di elaborazione reale.

2. Creazione del filtro con un'azione di registrazione

Un altro approccio sicuro e consigliato consiste nell'implementare il filtro senza interrompere le attività, ad esempio la registrazione, anziché applicare un'azione aggressiva come la perdita, il rimbalzo o la messa in quarantena dei messaggi.

Configurando il filtro per registrare una voce quando viene trovata una corrispondenza, gli amministratori possono:

- Osserva frequenza di abbinamento
- Rileva trigger imprevisti
- Convalida dell'impatto sulle prestazioni
- Analizzare il comportamento reale del traffico

Questo approccio inserisce efficacemente il filtro in una fase di monitoraggio controllato all'interno del traffico di produzione. Una volta completata una convalida sufficiente e verificato che il comportamento è corretto, l'azione può essere impostata in modo sicuro sulla modalità di imposizione.

Introduzione dell'espressione in un filtro contenuto e in un dizionario

Dopo aver progettato e convalidato correttamente l'espressione regolare, il passo successivo è capire come inserirla in Cisco ESA. La sintassi può apparire leggermente diversa a seconda che l'espressione sia configurata direttamente in una condizione di filtro contenuto o all'interno di un dizionario. Questa differenza causa spesso confusione.

Introduzione dell'espressione in un filtro contenuto

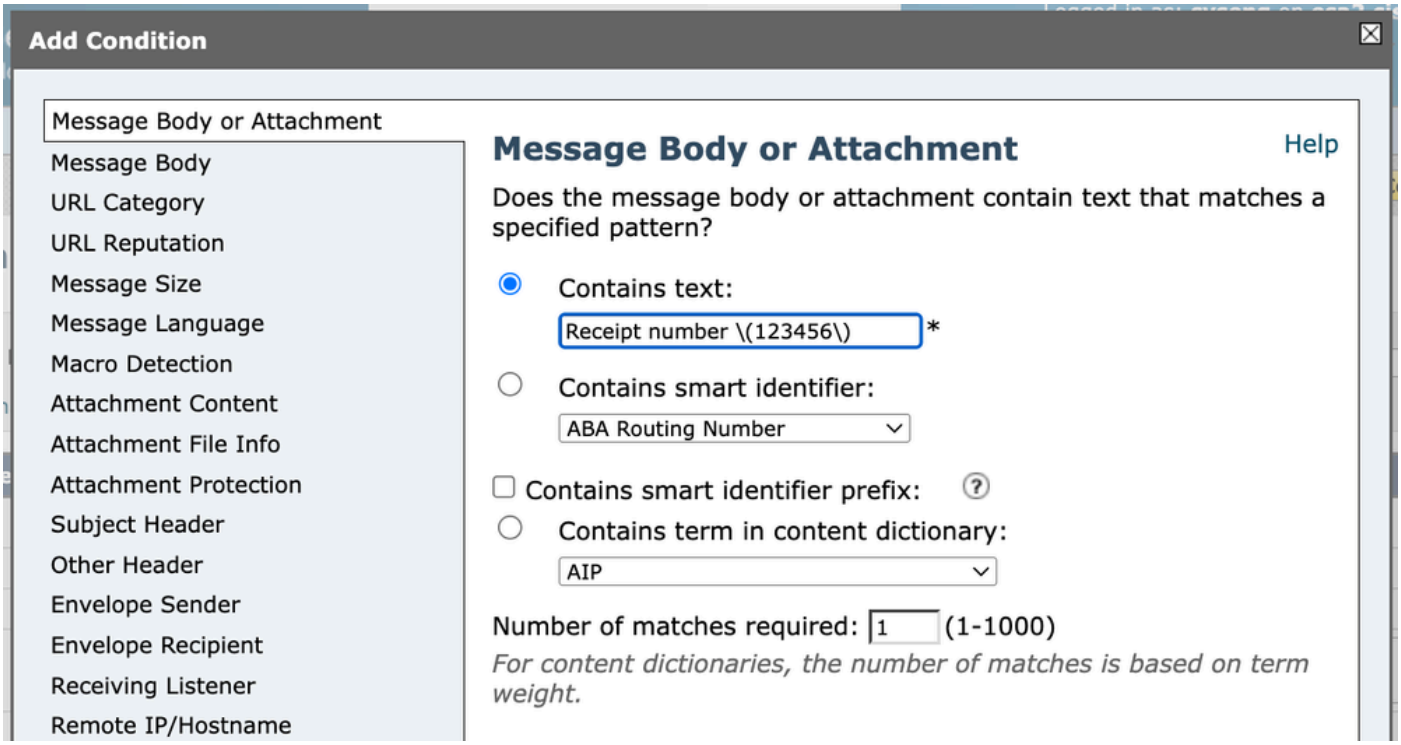
Quando si configura una condizione del filtro contenuto, ad esempio corrispondente all'intestazione Oggetto, è necessario immettere l'espressione regolare nel campo della condizione. Se si desidera trovare una corrispondenza con il testo:

```
Receipt number (123456)
```

Dobbiamo evitare le parentesi perché sono caratteri speciali nelle espressioni regolari.

Pertanto, il regex stesso deve essere scritto come segue:

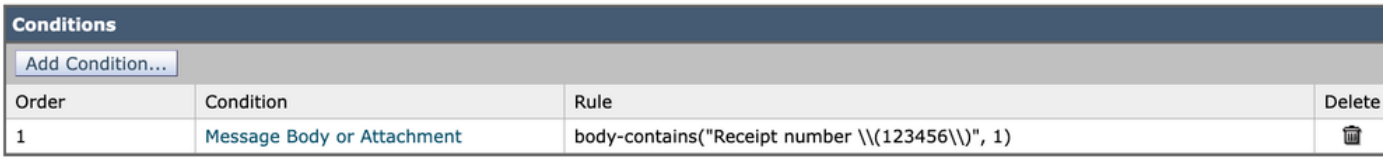
```
Receipt number \(123456\)
```



Filtro contenuti 1

Tuttavia, quando si visualizza la condizione di filtro completa nell'output della GUI o della configurazione avanzata, può apparire come:

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



Filtro contenuti 2

A prima vista, ciò può creare confusione. La doppia barra rovesciata (\\) è dovuta al fatto che la barra stessa è anche un carattere speciale all'interno delle stringhe racchiuse tra virgolette. In

questo contesto, una barra rovesciata viene utilizzata per uscire dalla parentesi per il motore regex, mentre la seconda barra rovesciata viene utilizzata per uscire dalla barra rovesciata all'interno della stringa citata.

In termini pratici:

\(123456\) è l'espressione regolare effettiva.

\\(è il modo in cui il sistema rappresenta \(all'interno di una stringa di configurazione racchiusa tra virgolette.

Anche se appare diversa quando viene visualizzata, l'espressione regolare logica da valutare rimane:

Numero ricevuta \(123456\)

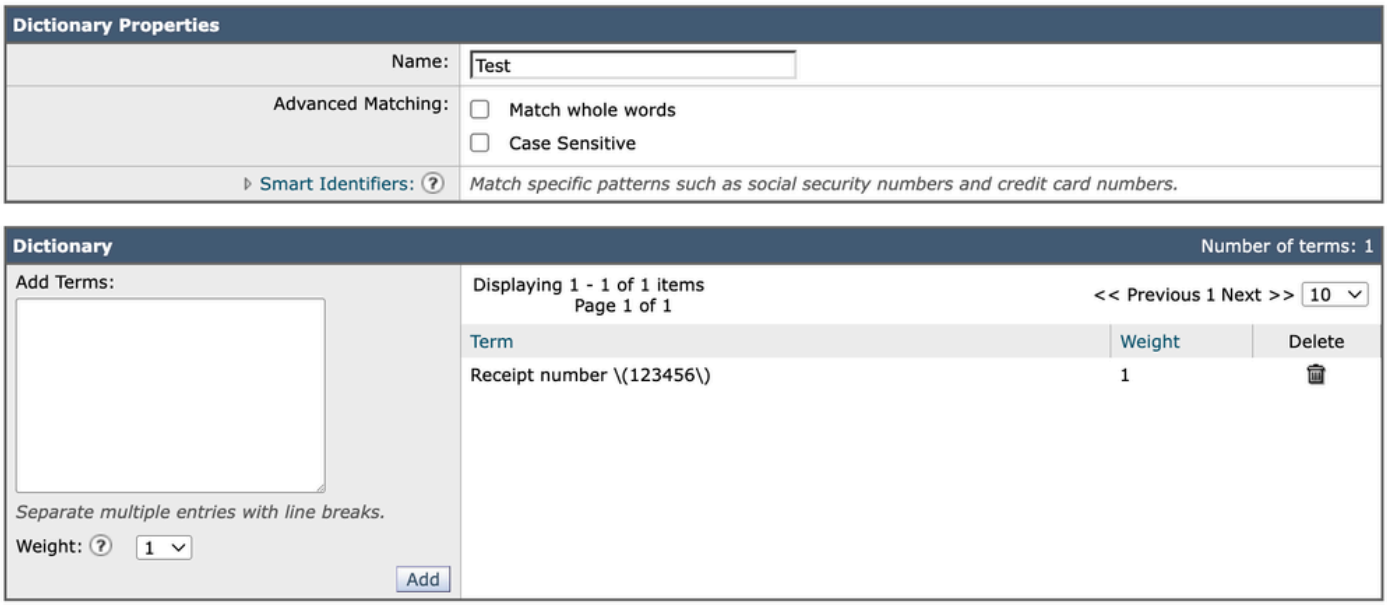
Si tratta semplicemente di un'operazione di escape di stringa nell'output di configurazione.

Introduzione all'espressione in un dizionario

Quando si aggiunge la stessa espressione a un oggetto Dictionary, la voce viene introdotta direttamente come:

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

In questo caso, continua a essere visualizzato esattamente come è scritto. A differenza della rappresentazione grafica di Content Filter, i dizionari non richiedono livelli di escape aggiuntivi nel formato di configurazione visiva.



Dizionario

Ogni voce del dizionario viene valutata come testo normale o come espressione regolare a seconda della sua struttura. Se vengono inclusi caratteri speciali, ad esempio le parentesi in questo caso, è necessario che l'espressione sia già stata inserita correttamente come carattere di escape.

Informazioni su "Parole intere"

Quando si configura un dizionario, è disponibile un'opzione denominata "Parole intere". In molti casi, si consiglia di non utilizzare questa impostazione quando si utilizzano espressioni regolari.

Il motivo è che il comportamento word-boundary può essere controllato in modo più preciso utilizzando gli ancoraggi regex.

Ad esempio:

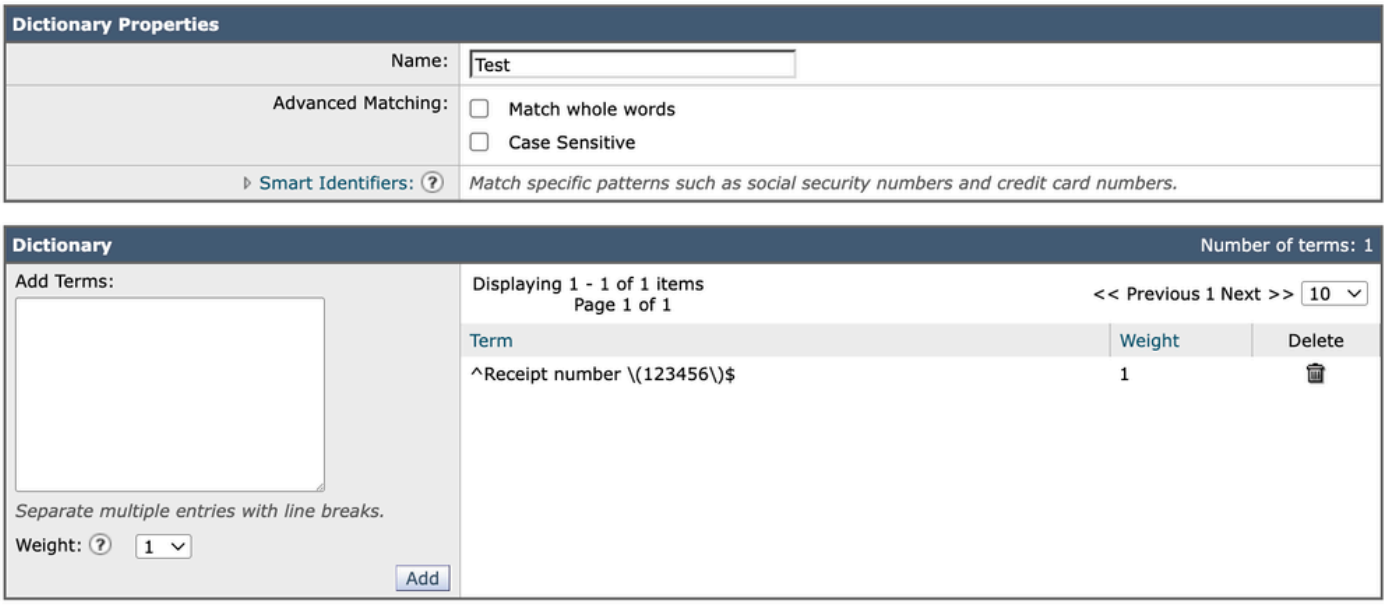
^ assicura che la corrispondenza inizi dall'inizio.

$ assicura che la corrispondenza termini alla fine.

Utilizzo di ancoraggi quali:

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

Fornisce un controllo esplicito e prevedibile del comportamento di corrispondenza esatta. Questo approccio evita potenziali ambiguità relative al modo in cui vengono interpretati i limiti delle parole, soprattutto in ambienti multilingue o non ASCII.



Dizionario 2

Per questo motivo, è in genere preferibile gestire la precisione delle corrispondenze direttamente

all'interno dell'espressione regolare anziché utilizzare l'opzione "Parole intere".

La comprensione di queste sottili differenze tra i filtri dei contenuti e i dizionari garantisce un comportamento coerente delle espressioni e riduce il rischio di errori di configurazione durante l'implementazione.

Classificazione costi Regex in Cisco ESA

Quando si utilizzano espressioni regolari in Cisco ESA, l'impatto sulle prestazioni dipende in gran parte dalla quantità di testo che deve essere analizzata dal motore e dalla quantità di backtracking che deve eseguire. Poiché l'ESA deve valutare interi corpi dei messaggi, parti MIME e persino allegati decodificati, i modelli inefficienti possono aumentare in modo significativo l'utilizzo della CPU.

Si tratta di una classificazione pratica dal costo di calcolo più alto a quello più basso.

Modelli ad alto rischio e più costosi

Queste espressioni possono avere un impatto significativo sulle prestazioni, in particolare sui messaggi di grandi dimensioni.

Quantificatori nidificati (caso peggiore)

Esempi:

```
(.*)+
(.+)+
(\S+)+
```

Questi sono estremamente pericolosi perché creano scenari di backtracking esponenziale.

Un quantificatore all'interno di un altro quantificatore forza il motore regex a provare molte combinazioni prima di fallire.

Nel traffico reale, ciò può causare picchi seri della CPU.

Consiglio: Evitare quantificatori nidificati non delimitati e ambigui.

Greedy .* Seguito da un modello obbligatorio

Esempio:

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

Questo criterio utilizza innanzitutto l'intero messaggio, quindi tiene traccia di nuovo carattere per carattere fino a trovare la sottostringa richiesta.

Se il modello non è presente, o appare vicino alla fine, il motore esegue il backtrack e verifica il token richiesto in molte posizioni, aumentando il costo della CPU.

In ESA, dove i corpi possono essere grandi e includere contenuto MIME, questo diventa molto costoso molto rapidamente.

Consiglio: non anteporre .* per rilevare le sottostringhe. L'ESA esegue già la ricerca nel contenuto valutato e i caratteri jolly principali aumentano solo il backtracking e l'utilizzo della CPU.

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

Alterazioni grandi con prefissi condivisi

Esempio:

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

Quando più alternative condividono la struttura, il motore valuta ciascuna diramazione in sequenza.

Se le prime diramazioni si avvicinano ma falliscono in ritardo, il motore riprova a lungo.

In questo modo i tempi di valutazione aumentano in modo significativo.

Costo Medio — Da Utilizzare Con Cautela

Questi modelli non sono catastrofici, ma possono essere ancora inefficienti.

Ampio .* Utilizzo

Esempio:

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

Anche se non esponenziale, .* consente comunque una corrispondenza illimitata. Se la sottostringa prevista non viene visualizzata rapidamente, il modulo analizza parti grandi del

messaggio.

Nell'ESA, questa condizione è comune quando si esegue la scansione dei corpi delle e-mail per rilevare gli URL di phishing.

Quantificatori Lazy (+?, *?)

Esempio:

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

I quantificatori lenti modificano la strategia di abbinamento (più breve). Possono ridurre il sovraccarico in alcuni modelli, ma nei carichi di lavoro di "ricerca" di grandi dimensioni possono aumentare i tentativi quando il token di terminazione è in ritardo o mancante.

In molti casi di utilizzo dell'ESA, non forniscono vantaggi reali e possono introdurre inutili tentativi interni.

Classi di caratteri molto generiche

Esempi:

```
\S+
.+
```

Ciò consente un'ampia gamma di corrispondenze, aumentando il numero di potenziali percorsi di backtracking.

È sempre preferibile utilizzare classi di caratteri più specifiche.

Basso costo: modelli sicuri ed efficienti

Questi sono consigliati per ambienti ESA di produzione.

Valori letterali fissi

Esempi:

```
text
iw\.adc
```

Le stringhe letterali sono le corrispondenze più efficienti possibili. Il motore esegue confronti diretti con un sovraccarico minimo.

Usa ancoraggi per limitare l'ambito di ricerca

Quando la corrispondenza è prevista in una posizione specifica, si consiglia di ancorare il pattern utilizzando ^ o $. Gli ancoraggi limitano la valutazione a posizioni fisse e impediscono al motore di analizzare l'intero contenuto inutilmente. In questo modo è possibile ridurre il backtracking e migliorare le prestazioni, in particolare nel corpo di un messaggio di grandi dimensioni o nelle intestazioni strutturate.

```
^Invoice$
```

Classi di caratteri specifiche

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

Queste funzionalità limitano il numero di risultati, riducendo notevolmente lo spazio di ricerca e limitando il backtracking.

Serie strutturate e vincolate

Esempio:

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

- Il dominio è fisso.
- Non utilizzare .*.
- non contiene modelli annidati catastrofici (ad esempio, (.*)+)
- Non sono necessari operatori pigri.
- Ogni sezione è vincolata.

Ciò riduce in modo significativo l'impatto sulla CPU rispetto alla corrispondenza di caratteri jolly.

Linee guida pratiche per Cisco ESA

Quando si progetta un regex per i filtri messaggi o contenuti:

1. Più specifico è il modello, migliori saranno le prestazioni.
2. Evitare .* a meno che non sia realmente necessario, e soprattutto evitare di posizionare i token necessari dopo di esso.
3. Non utilizzare mai quantificatori nidificati.
4. Preferire le classi di caratteri espliciti ai caratteri jolly.
5. Verificare sempre le nuove espressioni in modalità di monitoraggio prima dell'imposizione.

Confronto prestazioni Regex (contesto Cisco ESA)

Motivo	Consigliato	Rischio di backtracking	Impatto ESA	Alternativa consigliata
https?:\/\/.*/\?testo.*	No	Alta	Superiore	^https?:\/\/[A-Za-z0-9.-]+(?:\/[^?\s]*)*\/\?testo
https?:\/\/.*\?testo	Prestare attenzione	Medio-alta	Medio-alta	^https?:\/\/[^?\s]+\?testo$
https?:\/\/.*	No	Medio-alta	Media	^https?:\/\/[A-Za-z0-9.-]+(?:\/[^\s]
.*password	No	Alta	Superiore	password$
.*testo.*	No	Alta	Superiore	testo
.*(fattura|pagamento|trasferimento)	No	Alta	Superiore	(fattura|pagamento|trasferimento
(.+)+	Mai	Molto alta (esponenziale)	Severe (Grave)	Ristruttura senza quantificatori nidificati (esempio .+)
.*@.*	No	Alta	Superiore	[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}
\S+?	Non ideale	Media	Media	\S+ o classe più specifica come [A-Za-z0-9.-]+
.*\/amministratore	No	Alta	Superiore	\/admin$
.*(login|verifica).*	No	Alta	Superiore	(login|verifica)

^.*testo	No	Alta	Superiore	text$ (o ^testo se la posizione è importante)

Conclusioni

Le espressioni regolari sono uno strumento potente e flessibile all'interno di Cisco ESA, che consente un'ispezione accurata dei contenuti e un'applicazione avanzata delle policy sia nei filtri messaggi che nei filtri contenuti. Tuttavia, con questa flessibilità si arriva alla responsabilità. Espressioni progettate in modo errato o testate in modo inadeguato possono causare falsi positivi, mancati rilevamenti, peggioramento delle prestazioni o interruzione non intenzionale del traffico di posta elettronica legittimo.

Per questo motivo, l'uso di espressioni regolari in ESA deve sempre essere un approccio strutturato e disciplinato. La fase di creazione deve garantire che l'espressione sia sintatticamente corretta, correttamente evasa, efficiente e allineata logicamente all'obiettivo desiderato. Gli strumenti esterni e la generazione assistita dall'IA possono accelerare in modo significativo questo processo, ma non devono mai sostituire un'attenta convalida.

Altrettanto importante è la fase di convalida all'interno dell'ambiente ESA stesso. Poiché l'ESA elabora i messaggi tramite l'analisi MIME, la normalizzazione delle intestazioni e la decodifica dei contenuti, il comportamento reale può differire dalle aspettative teoriche. L'utilizzo di strumenti quali Traccia e distribuzione di filtri inizialmente in modalità di registrazione o monitoraggio consente agli amministratori di confermare il comportamento corretto senza rischi operativi.

In sintesi, le espressioni regolari devono essere mantenute il più possibile semplici, testate accuratamente e distribuite con cautela. Un filtro ben progettato e convalidato non solo applica le regole in modo efficace, ma protegge anche la stabilità del sistema e garantisce un comportamento prevedibile negli ambienti di produzione.

Documentazione

Per ulteriori dettagli tecnici e linee guida ufficiali su come le espressioni regolari vengono implementate e usate in Cisco ESA, gli amministratori devono consultare la documentazione del prodotto Cisco

La sezione "Espressioni regolari nelle regole" fornisce una panoramica della modalità di valutazione delle espressioni regolari all'interno dei filtri messaggi e dei filtri contenuti, incluse considerazioni sulla sintassi e sull'utilizzo nelle condizioni delle regole.

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 sezione "Linee guida per l'utilizzo delle espressioni regolari" offre consigli pratici sulla sintassi corretta, sull'ancoraggio delle espressioni, sulla gestione dei caratteri speciali e sull'eliminazione degli errori più comuni che possono influire sulle prestazioni o sulla precisione della

corrispondenza.

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

La revisione di queste risorse ufficiali è fortemente consigliata quando si progettano o si risolvono filtri che si basano sulle espressioni regolari, in quanto forniscono indicazioni autorevoli allineate alla versione AsyncOS specifica in uso.

Informazioni su questa traduzione

Cisco ha tradotto questo documento utilizzando una combinazione di tecnologie automatiche e umane per offrire ai nostri utenti in tutto il mondo contenuti di supporto nella propria lingua. Si noti che anche la migliore traduzione automatica non sarà mai accurata come quella fornita da un traduttore professionista. Cisco Systems, Inc. non si assume alcuna responsabilità per l’accuratezza di queste traduzioni e consiglia di consultare sempre il documento originale in inglese (disponibile al link fornito).