在Cisco ESA和CES中配置和驗證正規表示式

目錄

簡介

本檔案介紹ESA和CES如何在過濾器中使用正規表示式、關鍵行為差異以及在實施前進行測試的必要性。

背景資訊

本文檔介紹在郵件過濾器和內容過濾器中使用思科郵件安全裝置(ESA)和思科雲郵件安全(CES)如何處理正規表示式。它特別側重於瞭解正規表示式在這些元件中的行為方式，以及它們如何與電子郵件標頭、正文內容和附件互動。

必須從頭開始說明DLP模組中使用的正規表示式引擎的行為不同。因此，本文檔中描述的所有內容都只應用於郵件過濾器和內容過濾器，而不適用於DLP策略。

在ESA中使用正規表示式時，管理員必須瞭解電子郵件內容的計算方式與郵件客戶端中的直觀顯示方式不同。電子郵件包含信封資訊、結構化信頭、MIME部分和可能經過編碼的內容。因此，如果消息結構和正規表示式行為不完全被理解，過濾器執行的比較可能會產生意想不到的結果。

因此，在執行之前，任何使用正規表示式的新過濾器始終可以在監控模式下啟用。這樣可以對實際流量進行驗證，並防止出現意外的阻止或效能影響。

詞典和搜尋詞

建立郵件過濾器或內容過濾器時，在許多條件中輸入的術語將解釋為正規表示式。這是一個關鍵概念：即使管理員打算匹配文字文本，ESA也可以使用regex邏輯處理輸入。

這並不統一應用於所有條件型別。例如，當在某些結構化條件下搜尋特定IP地址時，該值不會解釋為正規表示式。但是，當在Subject標頭、郵件正文、特定標頭欄位或附件檔名中進行搜尋時，該值通常被視為正規表示式模式。

一個常見的例子清楚地說明了這一點。假設目標是阻止主題為的電子郵件：

```
Receipt number (123456)
```

由於括弧是正規表示式中的特殊字元（用於分組），因此必須將其轉義。

正確的表述是：

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

如果括弧未轉義，正規表示式引擎將它們解釋為分組運算子，而不是文字字元。根據模式，這可能導致意外的匹配或比預期不同的行為。

因此，必須瞭解哪些字元在正規表示式中具有特殊含義，並確保在需要文字匹配時正確轉義它們。

特殊字元及其轉義語法示例

第一列顯示包含特殊字元的示例文本，第二列顯示如何編寫正確的正規表示式語法以與Cisco ESA中的文本文本匹配(Python-style regex)。

要匹配的文字文本	正確正規表示式語法
收據編號(123456)	收據編號\(123456\)
user@example.com	user@example\.com
www.test.abc	www\.test\.abc
file_name.txt	file_name\.txt
price is 10.50	價格是10\.50
C:\Users\Admin	C:\\Users\\Admin
[機密]	\[機密\]
{invoice}	\{invoice\}
+34 600 123 456	\+34 600 123 456
問題?	問題？
100%保證	100%有保證（%不需要轉出）
星號*符號	星號*符號
A|B	A|B
插入符號^start	插入符號\^開始
100美元	美元\$100

限制正規表示式的使用

正規表示式必須謹慎使用，並且僅在必要時使用。雖然它們提供強大的匹配功能，但設計不當或設計不當的表達式會增加消息處理時間並產生意想不到的匹配。

需要注意的一個特殊構造是。* ，表示「任何字元，零次或更多次」。 如果放置在表達式的開頭或結尾，可能會導致過多的回溯和不必要的處理開銷。

思科文檔指出，在開頭或結尾使用.*的條目可能會導致系統在匹配特定MIME部件時在某些條件下鎖定。因此，思科建議儘可能避免使用前導或尾隨.*功能。

在許多情況下，當管理員可以簡單寫入發票並在ESA中產生相同的實際結果時，會使用.*invoice.*等模式。由於掃描引擎已經搜尋了相關內容區域，因此圍繞.*的字常常是冗餘且計算效率低下的。

⚠ 注意：一般建議是儘可能保持正規表示式簡單和精確。

郵件過濾器、內容過濾器和字典

Cisco ESA提供多種機制來評估報文和應用操作。消息過濾器在管道的開頭運行，並使用指令碼樣式的語法。它們非常靈活，允許涉及信封資料、信頭和附件屬性的高級邏輯。但是，由於消息過濾器在處理鏈早期執行，效率低下可能會對效能造成負面影響。

內容過濾器通過圖形介面配置，並在消息被接受後運行。對於大多數內容檢測使用情形，從效能的角度來看，內容過濾器更易於管理且更安全。

在郵件過濾器和內容過濾器中，正規表示式可以直接在條件中引入，也可以通過使用詞典間接引入。

詞典允許管理員集中使用可重複使用的搜尋詞語。每個條目都寫在單獨的行上，可以是純文字檔案或正規表示式。詞典也支援非ASCII字元，使其適合多語言環境。

某些情況下，某些複雜的正規表示式結構在詞典中無法具有相同的行為。發生這種情況時，正規表示式必須直接置於過濾器條件中，而不是放在字典中。

Cisco ESA允許建立最多150個內容詞典。預設情況下，可以配置100個詞典，除非使用dictionaryconfig命令通過CLI修改限制。

詞典還可以執行術語加權。每個術語可以分配一個數字權重，當ESA掃描消息時，它會將該術語的出現次數乘以其權重。將所得得分與過濾器中定義的閾值進行比較。該評分模型允許更靈活且分級的策略實施。

此外，詞典可以包括智慧識別符號，它是用於結構化數字模式（如社會保險號碼或銀行識別符號）的演算法檢測器。

正規表示式引擎

Cisco ESA使用基於Python re模組樣式的正規表示式。雖然這提供了與通用Python regex語法的相容性，但ESA不一定支援完整Python環境中支援的每個高級功能。

為了進行精確的字串匹配，表達式必須在開頭使用^並在結尾使用$定位。如果沒有這些錨點，正規表示式引擎可以匹配子字串而不是完整值。

例如，表達式：

```
sun.com
```

匹配字串，例如：

```
thegodsunocommando
```

但是，表達式：

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

僅匹配準確字串sun.com。

當匹配空字串時，不要使用「」非常重要，因為這實際上匹配了所有字串。相反，正確的表達式是：

```
^$
```

由於Cisco ESA使用Python風格的正規表示式，因此有幾種方法執行不區分大小寫的比較。

預設情況下，如前所述，正規表示式區分大小寫。這意味著搜尋：

```
foo
```

只匹配foo，但不匹配FOO、Foo或fOo。

如果要執行不區分大小寫的匹配，可以在正規表示式開頭使用內聯標誌(?i)。這指示regex引擎忽略模式其餘部分的大小寫。

舉例來說：

```
(?i)foo
```

此表達式匹配：

- foo
- FOO
- Foo
- O

如果要完全匹配整個字串（忽略大小寫），可以將不區分大小寫的標誌與錨點組合：

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

這確保了全部價值完全為「foo」，不管資本化如何。

另一個（不太實際的）替代方法是使用字元類來明確定義所有可能的組合，例如：

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

但是，這種方法會變得難以維護，建議不要使用(?i)標誌來代替。

在大多數ESA場景中，區分大小寫匹配的首選和最乾淨方法是使用：

```
(?i)
```

在正規表示式開頭。

非ASCII字元和字邊界

在使用雙位元組字符集的語言中，詞邊界或大小寫的概念不能按預期運行。當編碼或區域設定未知時，依賴於\w等結構的複雜表達式會產生不一致的結果。

在這種情況下，建議禁用詞典配置中的詞邊界實施，或簡化表達式，以避免依賴於不明確的字元類。

使用非ASCII詞典時，CLI顯示無法根據終端編碼正確呈現字元。在這種情況下，建議將詞典匯出到文本檔案，對其進行外部編輯並重新匯入。

寫入高效過濾器

在寫入篩選器時，效率至關重要，尤其是在高容量環境中。一個常見的錯誤是為類似的匹配項編寫長的OR條件鏈。

例如，單獨檢查數十個附件擴展會強制重新設計引擎重複初始化。這會增加CPU使用率並降低可維護性。

使用單個正規表示式中的交替方式將比較分組，而不是編寫許多單獨的比較，這樣可以大大減少處理開銷。這減少了呼叫regex引擎的次數，並使過濾器更易於維護。

高效的過濾器設計不僅關乎可讀性，還直接影響系統效能。

PDF和正規表示式

根據PDF的生成方式，匹配PDF檔案中的內容可能會產生意外的結果。某些PDF在其內部表示中不包含邏輯空格或換行符。掃描引擎嘗試基於字詞定位來重構邏輯間距。

如果使用多種字型或字型大小構建單詞，則內部表示可以對文本進行分段。例如，「callout」一詞

在內部可以解釋為「call out」或「c a l lout」。

在這種情況下，嘗試匹配表達式「callout」可能會失敗，因為內部表示法不包含該完全連續的字串。管理員在設計針對PDF附件的基於內容的策略時必須注意這一限制。

測試正規表示式

在將正規表示式部署到生產環境之前對其進行測試是一項關鍵操作要求。當根據實際電子郵件流量進行評估時，語法正確的正規表示式的行為可能截然不同。如果沒有正確的測試，過濾器可能會生成誤報、無法檢測預期模式、引入效能開銷或無意中斷合法郵件流。

測試必須採用結構化的兩階段流程，以便在生產中啟用過濾器之前將風險降至最低。

階段1 — 正規表示式設計和驗證
第一階段的重點是設計和驗證正規表示式本身，然後再將其整合到Cisco ESA中。

1.使用regex101或類似工具

線上平台(如http://regex101.com)在設計階段非常有用。使用這些工具時，必須選擇Python風格來近似ESA的regex引擎。

這些平台使管理員能夠：

- 驗證語法正確性
- 確認特殊字元已正確轉義
- 測試匹配和非匹配案例
- 視覺化分組和量詞行為
- 識別潛在的貪婪結構，例如。*

但是，這些工具模擬標準Python regex行為，可支援未在Cisco ESA中完全實施的功能。因此，它們必須被視為初步的驗證工具，而不是明確的相容性測試。

2.使用AI模型(ChatGPT、Copilot、...)

基於AI的助理可以加速正規表示式(regex)的建立，特別是在複雜的匹配場景中。通過以自然語言描述期望的行為，管理員可以獲得初始正規表示式提議，然後可以對其進行細化。

AI工具尤其有幫助：

- 生成複雜的分組表達式
- 將業務要求轉換為regex語法
- 將長期OR條件簡化為分組變更

然而，必須始終以批判的眼光來審視人工智慧產生的表達方式。它們可能帶來效率低下、不支援的結構或過於複雜的邏輯。人工智慧援助必須被視為起草援助，而不是最終驗證。每個AI生成的表達式仍必須使用結構化驗證方法進行測試。

第2階段 — 思科ESA中的過濾器行為驗證

一旦表達式本身經過驗證，第二階段將重點確認其在應用於實際消息處理時在Cisco ESA中的行為。

1.使用CES控制檯中的跟蹤功能

Cisco Email Security(CES)控制檯中的Trace（跟蹤）功能允許管理員模擬和分析特定郵件的處理方式。這是在執行前驗證過濾器行為的最可靠方法之一。

跟蹤提供以下內容的可視性：

- 消息的解析方式
- 評估哪些篩選器
- 是否觸發條件
- 規則執行的順序

由於ESA執行MIME解析、報頭規範化和內容解碼，裝置內部的行為可能與外部正規表示式測試工具有所不同。如需詳細說明，管理員必須查閱思科的官方檔案：

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

使用跟蹤可確保過濾器在真實處理引擎中的行為與預期一致。

2.使用日誌記錄操作建立過濾器

另一種安全且推薦的方法是使用無中斷操作（如日誌記錄）部署過濾器，而不是使用主動操作（如丟棄、退回或隔離郵件）。

通過將過濾器配置為在匹配時記錄條目，管理員可以：

- 觀察匹配頻率
- 檢測意外觸發器
- 驗證效能影響
- 分析實際流量行為

此方法有效地將過濾器放置在生產流量內的受控監控階段。一旦完成足夠的驗證並確認行為正確

，就可以安全地將該操作更改為實施模式。

在內容過濾器和詞典中引入表達式

正確設計和驗證正規表示式後，下一步是瞭解必須如何將其輸入到Cisco ESA中。根據表達式是直接在內容過濾器條件中配置還是在詞典中配置，語法可能會略有不同。這種差異常常造成混淆。

在內容過濾器中引入表達式

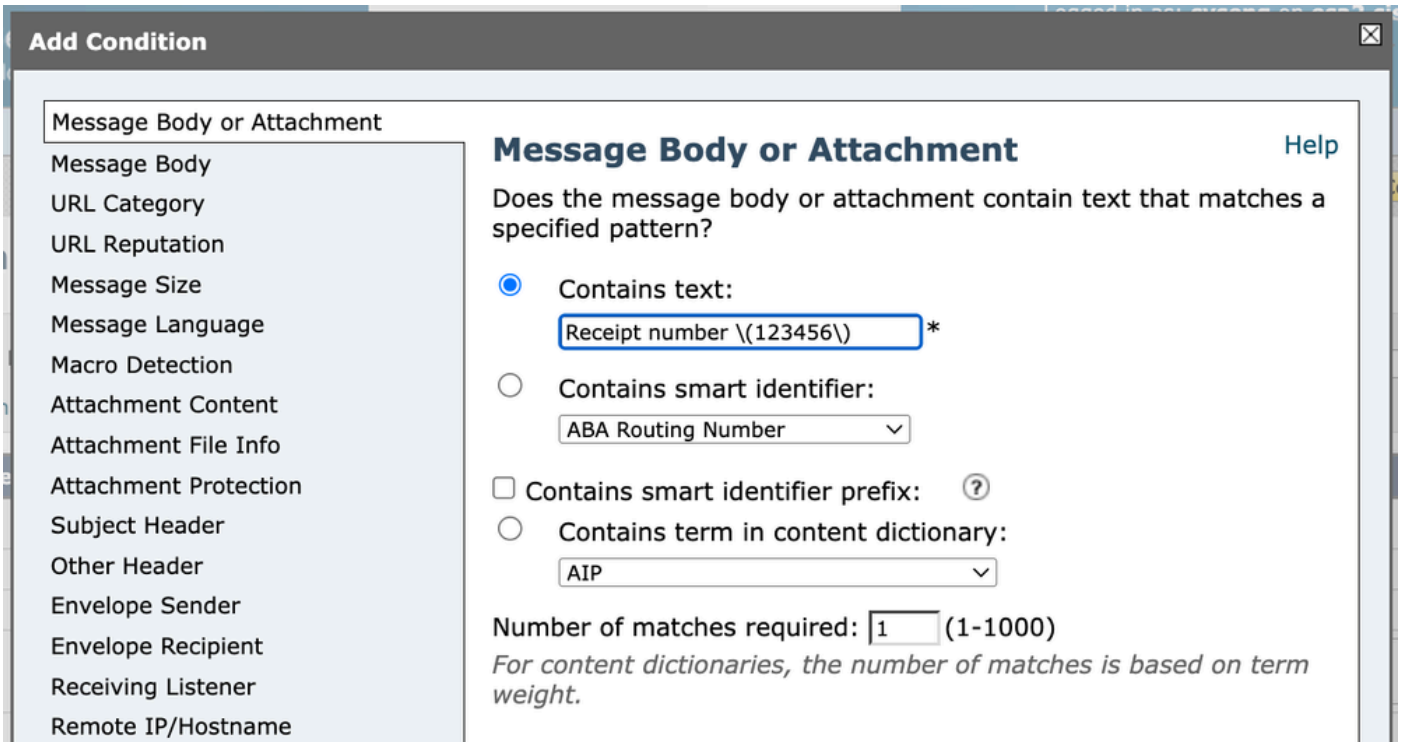
配置內容過濾器條件（例如，匹配主題標頭）時，必須在條件欄位中輸入正規表示式。如果要匹配文字文本：

```
Receipt number (123456)
```

必須轉義括弧，因為它們是正規表示式中的特殊字元。

因此，正規表示式本身必須寫為：

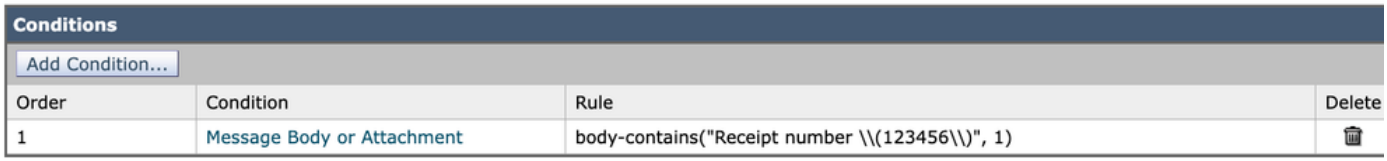
```
Receipt number \(123456\)
```



內容過濾器1

但是，當在GUI或高級配置輸出中檢視完全過濾條件時，它可能顯示為：

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



內容過濾器2

乍一看，這可能令人困惑。雙反斜槓(\\)的原因在於反斜槓本身也是帶引號字串中的特殊字元。在此上下文中，一個反斜線用於轉義regex引擎的括弧，而第二個反斜線用於轉義帶引號的字串中的反斜線。

從實際角度來說：

\(123456\)是實際的正規表示式。

\\()是系統在帶引號的配置字串內表示\()的方式。

儘管顯示時它顯示不同，但所計算的邏輯正規表示式將保留：

收據編號\(123456\)

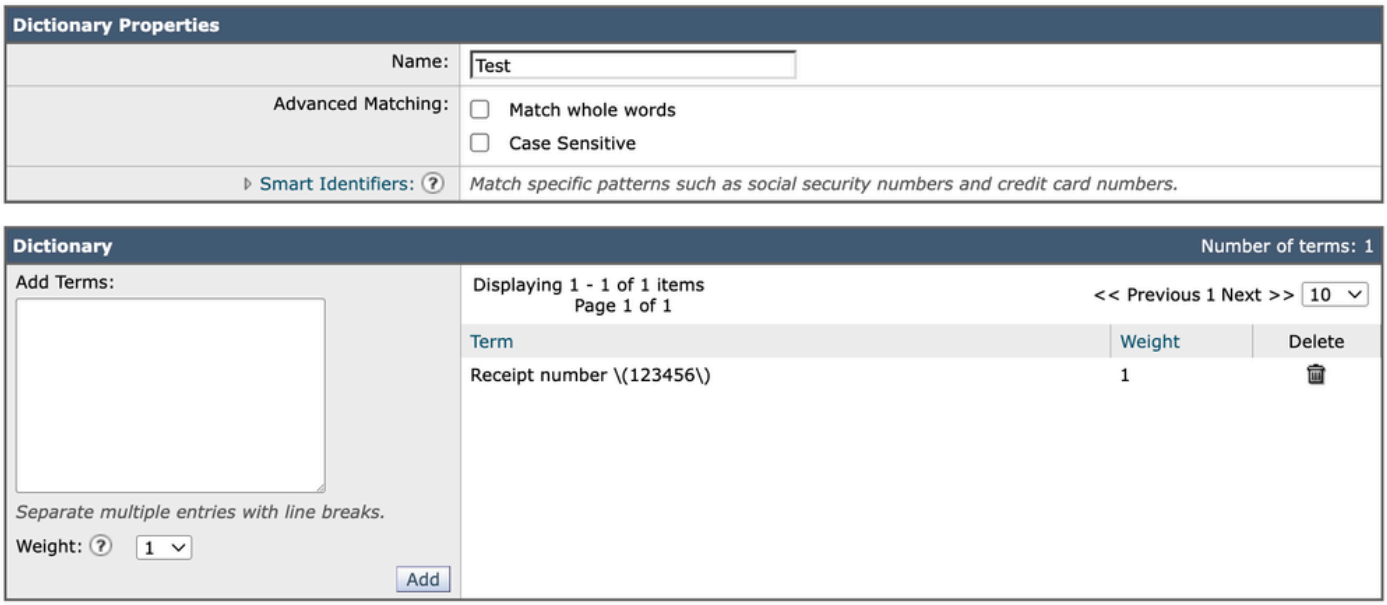
這只不過是配置輸出中轉義的字串問題。

在詞典中引入表達式

將同一表達式新增到詞典時，該條目直接引入為：

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

在這種情況下，它會繼續完全按照寫入的內容顯示。與內容過濾器GUI表示法不同，詞典不需要其可視配置格式的附加轉義層。



字典

根據詞典的結構，每個詞典條目都計算為純文字檔案或正規表示式。如果包含特殊字元（本例中用括弧括起來），則在輸入表達式時必須已正確轉義。

關於"整字匹配"

在配置字典時，有一個名為「匹配整字」的選項。 在許多情況下，建議在處理正規表示式時不要依賴此設定。

原因是可以使用正則錨點更精確地控制字邊界行為。

舉例來說：

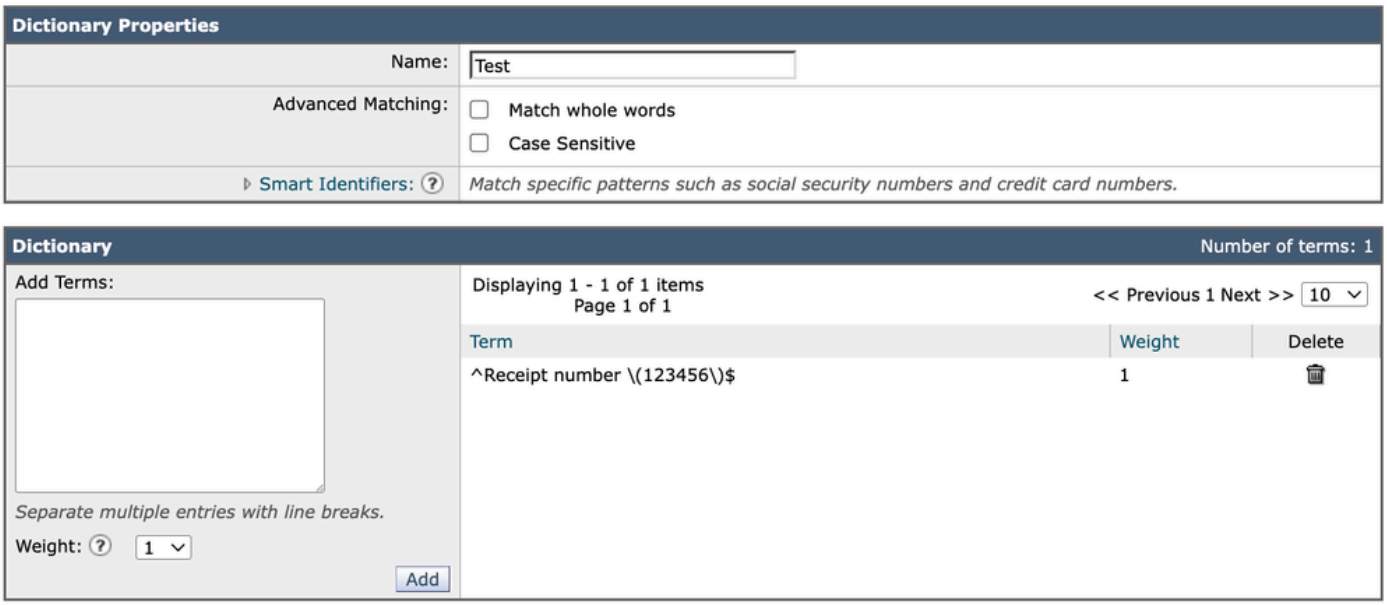
^確保匹配從開頭開始。

$確保匹配在結尾結束。

使用錨點，例如：

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

對精確匹配行為提供明確且可預測的控制。這種方法避免了與單詞邊界解釋方式相關的潛在歧義，特別是在多語言或非ASCII環境中。



字典2

因此，通常最好直接在正規表示式內管理匹配精度，而不是依賴「匹配整字」選項。

瞭解內容過濾器和詞典之間的這些細微差異可確保表達式的行為保持一致，並降低實施過程中出現配置錯誤的風險。

思科ESA的Regex成本排名

在Cisco ESA中使用正規表示式時，效能影響在很大程度上取決於引擎必須掃描多少文本以及必須執行多少回溯。由於ESA必須評估整個郵件正文、MIME部分，甚至已解碼附件，低效模式可能會顯著增加CPU使用率。

它是從最高計算成本到最低的實際排序問題。

最昂貴 — 高風險模式

這些表達式可能會嚴重影響效能，尤其是對於大型郵件。

巢狀限定符（最壞情況）

範例：

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

這些都是極其危險的，因為它們造成了指數級的回溯情景。

另一個量詞內的量詞強制正規表示式引擎在失敗之前嘗試許多組合。

在實際流量中，這可能導致嚴重的CPU峰值。

建議：避免使用無界且模糊的巢狀限定符。

貪婪。*後跟必需的模式

範例：

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

此模式首先使用整個消息，然後逐個字元回溯，直到找到所需的子字串。

如果模式不存在（或出現在末尾），引擎會回溯並在多個位置測試所需的令牌，這會增加CPU成本。

在ESA中，主體可能很大，並且包含MIME內容，因此成本會非常高。

建議：不要預先將。*用於檢測子字串。ESA已經搜尋了評估的內容，而領先的萬用字元只會增加回溯和CPU使用率。

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

具有共用首碼的大型替代專案

範例：

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

當多個備選方案共用結構時，引擎將按順序評估每個分支。

如果早期分支幾乎匹配，但故障較晚，引擎會大量重試。

這顯著增加了評估時間。

中等成本 — 謹慎使用

這些模式不是災難性的，但仍可能效率低下。

廣泛.*使用

範例：

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

雖然不是指數級匹配，.*仍允許無限匹配。如果預期的子字串沒有快速顯示，引擎將掃描大部分消息。

在ESA中，當掃描電子郵件正文查詢網路釣魚URL時，這種情況很常見。

延遲量詞(+?,*?)

範例：

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

延遲量詞更改匹配策略（最短優先）。 它們可以減少某些模式中的過度匹配，但在大型「搜尋」工作負載中，它們可以在終端令牌延遲或丟失時增加嘗試。

在許多ESA使用案例中，它們並不能提供真正的益處，並且可能會引入不必要的內部重試。

非常通用的字元類

範例：

```
\S+
.+
```

這些允許較大的匹配範圍，從而增加了潛在回溯路徑的數量。

更具體的字元類總是更可取。

低成本 — 安全高效的模式

建議為生產ESA環境提供這些功能。

固定文字

範例：

```
text
iw\.adc
```

文字字串是最有效的匹配項。該引擎以最小的開銷執行簡單比較。

使用錨點限制搜尋範圍

當匹配預期位於特定位置時，請考慮使用^或$錨定模式。錨點將評估限制在固定位置，並阻止引擎不必要地掃描整個內容。這可減少回溯並提高效能，特別是在大型郵件正文或結構化信頭中。

```
^Invoice$
```

特定字元類

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

這些限制可以匹配的內容，從而顯著減少搜尋空間並限制回溯。

結構化模式和約束模式

範例：

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

- 域已修復。
- 不使用。*。
- 不包含災難性巢狀模式(例如，(.*)+)
- 沒有不必要的懶惰操作員。
- 每個截面都受約束。

與廣泛的萬用字元匹配相比，這顯著降低了CPU影響。

Cisco ESA實用指南

為郵件或內容過濾器設計正規表示式時：

1. 模式越具體，效能越好。
2. 避免使用。*，除非確實必要，尤其是避免在它之後放置所需的令牌。

3. 切勿使用巢狀的限定符。
4. 優先使用顯式字元類而不是萬用字元。
5. 實施之前，始終在監控模式下測試新表達式。

Regex效能比較（思科ESA環境）

模式	建議	回溯風險	ESA影響	建議的替代方案
https?:\/\/.*/\？文本。*	否	高	更高	^https?:\/\/[A-Za-z0-9.-]+(?:\/[^?\s]*)*\/\？文本
https?:\/\/.*\？文本	⚠慎使用	中高	中高	^https?:\/\/[^?\s]+\?text$
https?:\/\/.*	否	中高	中	^https?:\/\/[A-Za-z0-9.-]+(?:\/[^\s]*)*
.*密碼	否	高	更高	密碼$
.*文本。*	否	高	更高	文本
.*(invoice|payment|transfer)	否	高	更高	（發票|付款|轉移）$
(.+)+	從不	非常高（指數）	嚴重	不使用巢狀量化符進行重構（示例。+）
.*@.*	否	高	更高	[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}
\S+?	不理想	中	中	\S+或更具體的類，如[A-Za-z0-9.-]+
.*\/admin	否	高	更高	\/admin$
.*(login|verify)。*	否	高	更高	（登入|驗證）
^.*文本	否	高	更高	text$(或^text（如果位

				置重要）

結論

正規表示式是Cisco ESA中功能強大且靈活的工具，可在郵件過濾器和內容過濾器中實現精確的內容檢測和高級策略實施。然而，這種靈活性意味著責任。設計不佳或測試不充分的表達式可能導致誤報、漏檢、效能下降或無意中斷合法電子郵件流量。

因此，在歐空局中使用正規表示式必須始終是結構化和有條理的方法。建立階段必須確保表達式的語法正確、正確轉義、高效且與預期目標在邏輯上保持一致。外部工具和AI輔助生成可以大大加快這一過程，但它們絕不能取代仔細的驗證。

同樣重要的是歐空局環境本身的驗證階段。由於ESA通過MIME解析、報頭規範化和內容解碼處理郵件，因此實際行為可能與理論預期不同。管理員最初在日誌記錄或監控模式下使用諸如跟蹤和部署過濾器之類的工具可以確認正確的行為，而不會產生操作風險。

總之，正規表示式必須保持儘可能簡單，必須經過徹底測試，並且必須謹慎部署。設計完善且經過正確驗證的過濾器不僅可以有效地實施策略，還可以保護系統穩定性並確保生產環境中的可預見行為。

檔案

有關正規表示式在Cisco ESA中如何實施和使用的其他技術詳細資訊和官方指南，管理員必須查閱思科產品文檔

「規則中的正規表示式」一節概述如何在消息過濾器和內容過濾器內計算正規表示式，包括規則條件內的語法注意事項和用法。

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

「使用正規表示式的准則」一節提供了一些實用的建議，包括正確的語法、錨定表達式、處理特殊字元，以及避免可能影響效能或匹配準確度的常見錯誤。

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

在設計或排除依賴於正規表示式的過濾器時，強烈建議檢視這些官方資源，因為它們提供了與正在使用的特定AsyncOS版本一致的權威性指南。

關於此翻譯

思科已使用電腦和人工技術翻譯本文件，讓全世界的使用者能夠以自己的語言理解支援內容。請注意，即使是最佳機器翻譯，也不如專業譯者翻譯的內容準確。Cisco Systems, Inc. 對這些翻譯的準確度概不負責，並建議一律查看原始英文文件（提供連結）。