Linee guida per le espressioni regolari e considerazioni sulle prestazioni per il filtro URL

Sommario

Introduzione

In questo documento vengono descritte le linee guida e le considerazioni sulle prestazioni relative all'utilizzo delle espressioni regolari nel filtro URL con il motore UTD. Il filtro URL nel motore UTD utilizza la libreria di espressioni regolari PCRE2.

Contributo di Eugene Khabarov, Cisco Engineering.

Prerequisiti

Requisiti

Cisco raccomanda la conoscenza dei seguenti argomenti:

- Sintassi delle espressioni regolari (regex)
- Nozioni base sul filtro URL
- Configurazione Unified Threat Defense (UTD)
- Differenze tra i protocolli HTTPS e HTTP

Componenti usati

Il documento può essere consultato per tutte le versioni software o hardware.

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata ripristinata ai valori predefiniti. Se la rete è operativa, valutare attentamente eventuali conseguenze derivanti dall'uso dei comandi.

Premesse

Mentre PCRE2 è potente, alcune espressioni complesse o "greedy" possono causare un eccessivo backtracking e possono raggiungere i limiti interni nel motore regex. In questo caso, l'elaborazione di un modello può richiedere troppo tempo e alla fine viene considerata come 'nessuna corrispondenza'.

Punti chiave

- PCRE2 applica limiti interni alle operazioni di backtracking o al tempo di abbinamento per proteggere le risorse del sistema.
- Alcuni modelli sono sintatticamente validi ma computazionalmente non sicuri e possono attivare il 'backtracking catastrofico'.
- Quando questi limiti vengono superati, il motore regex può interrompere l'elaborazione e non restituire alcuna corrispondenza, anche se l'URL corrisponde logicamente al modello.

Modelli da evitare

Evitare i costrutti regex che combinano:

- Quantificatori nidificati, ad esempio: (...+)*, (.*)*, (.+)+, e così via
- Caratteri jolly (.) ripetuti su ampie parti della stringa, in particolare vicino alla fine del motivo
- Punti senza escape nei nomi di dominio se utilizzati insieme alla ripetizione

In questo caso, ad esempio, il modello è sintatticamente valido ma può risultare dispendioso da elaborare:

```
^([a-zA-Z0-9-]+.)*portal.example.com$
```

Nota: In questo caso, ([a-zA-Z0-9-]+.)* è un gruppo con un quantificatore nidificato (+ interno *) più un carattere jolly (.). Su alcuni input non corrispondenti, il motore regex può esplorare un numero molto elevato di percorsi di backtracking.

Procedure ottimali consigliate

Escape sempre punti nei nomi host

Utilizzare \. per trovare una corrispondenza con un punto letterale, ad esempio:

```
^([a-zA-Z0-9-]+\.)*portal\.example\.com$
```

Ancorare i pattern e limitare i caratteri

Utilizzare ^ e $e limitare ai caratteri previsti (ad esempio, [a-zA-Z0-9-] per le etichette dell'host) per ridurre il backtracking.

Evitare Ripetizioni Annidate E Senza Limiti, Se Possibile

Preferire costrutti più semplici piuttosto che modelli complessi che provano a coprire tutto in un regex. Si considerino diverse voci specifiche anziché un'espressione molto ampia.

Modelli di test in un tester compatibile con PCRE2

Prima della distribuzione, testare i modelli regex in un ambiente compatibile con PCRE2 ed evitare i modelli che generano "backtracking catastrofico" o avvisi simili.

Nota: Se un modello regex raggiunge i limiti interni del motore PCRE2, il motore di filtro URL può considerarlo come 'no match'. In questi casi, la classificazione degli URL rientra nella categoria o nella reputazione, non nel risultato del regex sulla lista bianca o sulla lista nera. I limiti esatti sono specifici dell'implementazione e possono variare da una release all'altra. È necessario progettare i regessi in modo conservativo.

Differenze nella corrispondenza URL per HTTP e HTTPS

Il motore UTD controlla gli URL in modo diverso per il traffico HTTPS e HTTP. Ciò influisce sul modo in cui le espressioni regolari devono essere progettate per il filtro URL.

Traffico HTTPS (TLS)

Per il traffico HTTPS crittografato, il motore UTD non decrittografa il payload per impostazione

predefinita.

- Il filtro URL utilizza l'indicazione del nome del server (SNI) fornita dal client TLS (Transport Layer Security).
- Il modello regex viene applicato solo al nome host SNI, ad esempio: api.example.com

In questo caso, un modello basato su nome host viene confrontato con la stringa api.example.com del nome host, ad esempio:

```
^([a-zA-Z0-9-]+\.)*example\.com$
```

Traffico HTTP (non crittografato)

Per il traffico HTTP normale, il motore UTD può visualizzare la richiesta HTTP completa (riga richiesta e intestazioni).

A seconda dell'implementazione, la stringa fornita al motore regex può includere:

- L'URL completo o la riga della richiesta (ad esempio, GET /path?param=value HTTP/1.1) oppure
- L'intestazione dell'host combinata con il percorso (ad esempio, api.example.com/path)

Di conseguenza, l'input regex per HTTP può contenere caratteri aggiuntivi quali /, ? e stringhe di query, non solo il nome host nudo.

Implicazioni della configurazione

Un regex progettato esclusivamente per i nomi host (ad esempio, solo api.example.com corrispondente) può corrispondere correttamente a HTTPS (SNI) ma non a una richiesta HTTP contenente un URL completo o una stringa host+percorso.

Per filtrare il traffico HTTP e HTTPS con lo stesso modello, è necessario:

- Modelli di progettazione principalmente intorno ai nomi host
- Verifica del comportamento rispetto a HTTP e HTTPS nei log UTD

Verifica

Abilita registrazione debug

Passaggio 1. Eseguire il comando debug utd engine standard url-filtering level info per abilitare la registrazione del debug.

Passaggio 2. Eseguire il comando show logging process ioxman module utd | includere il comando api.example.com per verificare i log.

Output di esempio:

```
2025/11/27 11:45:28.195000350 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF event->server_
2025/11/27 11:45:28.195001873 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF   URL: api.exa
2025/11/27 11:45:28.195009216 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF Regex matched
2025/11/27 11:45:28.195022442 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF URLF whitelist
2025/11/27 11:45:33.530605572 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF   URL: api.exa
2025/11/27 11:45:33.530606333 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF Regex not matc
2025/11/27 11:45:33.530614980 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF URLF whitelist
```

Esempi di configurazione

Corrispondenza basata su host

Per consentire tutti i sottodomini di example.com, utilizzare il modello basato su nome host (baseline) consigliato:

```
^([a-zA-Z0-9-]+\.)*example\.com$
```

Schema:

- Trova/sostituisce example.com, api.example.com, foo.bar.example.com e così via.
- È adatto per la corrispondenza HTTPS (SNI)
- Può corrispondere anche a HTTP se la stringa visualizzata dal motore è il nome host vuoto

Corrispondenza host/percorso HTTP

Se HTTP include host/percorso e si desidera ignorare il percorso, è possibile trovare la corrispondenza con il prefisso hostname e fare in modo che il regex si arresti in corrispondenza di un limite di parola anziché alla fine. *, ad esempio:

```
^([a-zA-Z0-9-]+\.)*example\.com\b
```

Nota: In questo caso, \b (limite parola) consente effettivamente l'utilizzo di caratteri quali / o ? per seguire il nome host senza richiedere un carattere jolly.* esplicito. Questo è generalmente più economico che aggiungere .* alla fine e si allinea meglio con le linee guida per evitare ulteriori caratteri jolly non delimitati.

Attenzione: La stringa esatta passata nel motore regex per le richieste HTTP è specifica

---

⚠ dell'implementazione e può evolversi. In caso di dubbi, verificare i modelli rispetto al traffico HTTP e HTTPS in un ambiente lab e verificare le corrispondenze nei log UTD prima di eseguire la distribuzione in produzione.

Informazioni correlate

- Guida alla configurazione della sicurezza di Cisco Catalyst SD-WAN, Cisco IOS XE Catalyst SD-WAN release 17.x
- Supporto tecnico Cisco e download

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).