Richtlinien für reguläre Ausdrücke und Leistungsüberlegungen für die URL-Filterung

Inhalt

Einleitung

In diesem Dokument werden die Richtlinien und Leistungsüberlegungen für die Verwendung regulärer Ausdrücke bei der URL-Filterung mit der UTD-Engine beschrieben. Bei der URL-Filterung im UTD-Modul wird die Bibliothek für reguläre PCRE2-Ausdrücke verwendet.

Gefördert von Eugene Khabarov, Cisco Engineering.

Voraussetzungen

Anforderungen

Cisco empfiehlt, dass Sie über Kenntnisse in folgenden Bereichen verfügen:

- Syntax regulärer Ausdrücke (regex)
- URL-Filterungskonzepte
- Unified Threat Defense (UTD)-Konfiguration
- HTTPS/HTTP-Protokollunterschiede

Verwendete Komponenten

Dieses Dokument ist nicht auf bestimmte Software- und Hardware-Versionen beschränkt.

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die möglichen Auswirkungen aller Befehle kennen.

Hintergrundinformationen

Während PCRE2 leistungsstark ist, können bestimmte komplexe oder 'gierige' Ausdrücke zu übermäßigem Backtracking führen und interne Grenzen in der Regex-Engine erreichen. In diesem Fall kann die Verarbeitung eines Musters zu lange dauern und wird letztendlich als "keine Übereinstimmung" behandelt.

Wichtigste Punkte

- PCRE2 setzt interne Beschränkungen für Rückverfolgungsschritte oder die Abgleichzeit durch, um Systemressourcen zu schützen.
- Einige Muster sind syntaktisch gültig, aber rechnerisch unsicher und können 'katastrophales Backtracking' auslösen.
- Wenn diese Grenzwerte überschritten werden, kann die Regex-Engine die Verarbeitung abbrechen und keine Übereinstimmung zurückgeben, selbst wenn die URL logisch mit dem Muster übereinstimmt.

Vermeidbare Muster

Vermeiden Sie Regex-Konstrukte, die Folgendes kombinieren:

- Verschachtelte Quantifizierer, z. B.: (...+)*, (.*)*, (.+)+ usw.
- Platzhalter (.), die sich über große Teile der Zeichenkette wiederholen, insbesondere am Ende des Musters
- Bei Verwendung mit Wiederholungen nicht entkommene Punkte in Domänennamen

Hier ist das Muster beispielsweise syntaktisch gültig, kann jedoch aufwändig zu verarbeiten sein:

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

Anmerkung: In diesem Fall ist ([a-zA-Z0-9-]+.)* eine Gruppe mit einem verschachtelten Quantifizierer (+ innerhalb von *) plus einem Platzhalter (.). Auf einigen nicht übereinstimmenden Eingaben kann die Regex-Engine eine sehr große Anzahl von Backtracking-Pfaden erkunden.

Empfohlene Best Practices

Immer Fluchtpunkte in Hostnamen

Verwenden Sie \., um einen literalen Punkt zu finden, z. B.:

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

Muster ankern und Zeichen einschränken

Verwenden Sie ^ und $ und beschränken Sie diese auf erwartete Zeichen (z. B. [a-zA-Z0-9-] für Hostbezeichnungen), um die Rückverfolgung zu reduzieren.

Vermeiden Sie verschachtelte, unbegrenzte Wiederholungen, sofern möglich

Bevorzugen Sie einfachere Konstrukte anstelle komplexer Muster, die alles in einem Regex abdecken wollen. Betrachten Sie mehrere spezifische Einträge anstelle eines sehr breiten Ausdrucks.

Testmuster in einem PCRE2-kompatiblen Tester

Testen Sie vor der Bereitstellung die Regex-Muster in einer PCRE2-kompatiblen Umgebung, und vermeiden Sie Muster, die ein "katastrophales Backtracking" oder ähnliche Warnungen auslösen.

Anmerkung: Wenn ein reguläres Muster die internen Grenzen des PCRE2-Moduls erreicht, kann es von der URL-Filterungsengine als "keine Übereinstimmung" behandelt werden. In solchen Fällen fällt die URL-Klassifizierung auf die Kategorie oder die Reputation zurück, nicht auf das Ergebnis der Whitelist/Blacklist-regulären Ausdrücke. Die genauen Limits sind implementierungsspezifisch und können sich zwischen den Releases ändern. Sie müssen Ihre Regulierungsvorgaben konservativ gestalten.

Unterschiede beim URL-Matching für HTTP und HTTPS

Die UTD-Engine überprüft URLs unterschiedlich auf HTTPS- und HTTP-Datenverkehr. Dies beeinflusst, wie reguläre Ausdrücke für die URL-Filterung konzipiert werden müssen.

HTTPS-Datenverkehr (TLS)

Bei verschlüsseltem HTTPS-Datenverkehr entschlüsselt die UTD-Engine die Nutzlast nicht standardmäßig.

- Bei der URL-Filterung wird die Servernamensangabe (SNI) des TLS-Client (Transport Layer Security) Hello verwendet.
- Das regex-Muster wird nur auf den SNI-Hostnamen angewendet, z. B.: api.example.com

In diesem Fall wird ein auf einem Hostnamen basierendes Muster mit der Hostnamen-Zeichenfolge api.example.com abgeglichen, z. B.:

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

HTTP-Datenverkehr (unverschlüsselt)

Für reinen HTTP-Datenverkehr kann die UTD-Engine die vollständige HTTP-Anforderung (Anforderungszeile und Header) anzeigen.

Je nach Implementierung kann die an das Regex-Modul übergebene Zeichenfolge Folgendes enthalten:

- Die vollständige URL oder Anforderungszeile (z. B. GET /path?param=value HTTP/1.1) oder
- Der Host-Header wird mit dem Pfad kombiniert (z. B. api.example.com/path)

Daher kann die regex-Eingabe für HTTP zusätzliche Zeichen wie /, ? und Abfragezeichenfolgen enthalten, nicht nur den reinen Hostnamen.

Auswirkungen auf die Konfiguration

Ein nur für Hostnamen entwickelter regulärer Ausdruck (z. B. nur api.example.com) kann HTTPS korrekt (SNI) abgleichen, jedoch nicht mit einer HTTP-Anfrage, die eine vollständige URL oder eine Host+Pfad-Zeichenfolge enthält.

Um HTTP- und HTTPS-Datenverkehr nach demselben Muster zu filtern, müssen Sie:

- Design-Muster primär für Hostnamen
- Überprüfung des Verhaltens gegenüber HTTP und HTTPS in den UTD-Protokollen

Überprüfung

Debug-Protokollierung aktivieren

Schritt 1: Führen Sie den Befehl debug utd engine standard url-filterlevel info aus, um die Debug-Protokollierung zu aktivieren.

Schritt 2. Führen Sie den show logging process ioxman module utd | den Befehl api.example.com zum Überprüfen der Protokolle ein.

Beispiel:

```
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
```

Konfigurationsbeispiele

Hostbasierte Zuordnung

Um alle Subdomänen von example.com zuzulassen, verwenden Sie dieses empfohlene Muster für Hostnamen (Baseline):

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

Dieses Muster:

- Entspricht beispiel.com, api.beispiel.com, foo.bar.beispiel.com usw.
- Für HTTPS-Abgleich (SNI) geeignet
- Kann auch mit HTTP übereinstimmen, wenn die vom Modul angezeigte Zeichenfolge der reine Hostname ist

HTTP-Host-/Pfadzuordnung

Wenn HTTP host/path enthält und Sie den Pfad ignorieren möchten, können Sie das Präfix für den Hostnamen anpassen und den Regex an einer Wortgrenze anhalten lassen, anstatt an einer nachgestellten. *, zum Beispiel:

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

Anmerkung: Hier erlaubt \b (Wortgrenze) effektiv Zeichen wie / oder ?, um dem Hostnamen zu folgen, ohne dass ein expliziter .* Platzhalter erforderlich ist. Dies ist im Allgemeinen billiger als das Hinzufügen von .* am Ende und passt sich besser an die Anleitung an, um

---

zusätzliche unbegrenzte Platzhalter zu vermeiden.

Vorsicht: Die exakte Zeichenfolge, die für HTTP-Anforderungen an das Regex-Modul übergeben wird, ist implementierungsspezifisch und kann weiterentwickelt werden. Testen Sie im Zweifelsfall Muster für HTTP- und HTTPS-Datenverkehr in einer Laborumgebung, und überprüfen Sie die Übereinstimmungen in den UTD-Protokollen, bevor Sie sie in die Produktion implementieren.

Zugehörige Informationen

- Cisco Catalyst SD-WAN Security Configuration Guide, Cisco IOS XE Catalyst SD-WAN Version 17.x
- Technischer Support und Downloads von Cisco

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.