Konfigurieren benutzerdefinierter Skripts auf CPAR 8.0

Inhalt

Einführung

In diesem Dokument wird beschrieben, wie Sie das Verhalten von Cisco Prime Access Registrar (CPAR) 8.0 mithilfe von Skripts und Erweiterungspunkten anpassen.

Voraussetzungen

Anforderungen

Cisco empfiehlt, über Kenntnisse in folgenden Bereichen zu verfügen:

- CPAR 8.0 Anwendung

Verwendete Komponenten

Die Informationen in diesem Dokument basieren auf den folgenden Software- und Hardwareversionen:

- CPAR 8.0 wird auf CentOS 6.5 64 Bit installiert

Die Informationen in diesem Dokument wurden von den Geräten in einer bestimmten Laborumgebung erstellt. Alle in diesem Dokument verwendeten Geräte haben mit einer leeren (Standard-)Konfiguration begonnen. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die potenziellen Auswirkungen eines Befehls verstehen.

Hintergrundinformationen

CPAR kann sowohl von internen als auch von externen Skripts geändert werden. Skripte können in C/C++/Java/TCL geschrieben werden. Skripte können verwendet werden, um die Verarbeitung von RADIUS-, TACACS- und DIAMETER-Paketen zu ändern. Skripte können in CPAR in

Erweiterungspunkten referenziert werden. Erweiterungspunkte sind Einstellungen/Attribute, die unter einigen Konfigurationselementen angezeigt werden und auf ein Skript verweisen können. Laut Referenzhandbuch ist CPAR nicht für Datenverluste, Schäden usw. verantwortlich, die durch benutzerdefinierte Scripts verursacht werden.

Im Folgenden sehen Sie ein Beispiel für zwei Erweiterungspunkte in der Konfiguration von Netzwerkgeräten.

```
[ //localhost/Radius/Clients/piborowi ]
    Name = piborowi
    Description =
    Protocol = tacacs-and-radius
    IPAddress = 192.168.255.15
    SharedSecret = <encrypted>
    Type = NAS
    Vendor =
    IncomingScript~ =                          // Extension point for incomming traffic
    OutgoingScript~ =                          // Extension point for outgoing traffic
    EnableDynamicAuthorization = FALSE
    NetMask =
    EnableNotifications = FALSE
    EnforceTrafficThrottling = TRUE
```

Dem CPAR-Administrationsleitfaden zufolge gibt es mehrere verfügbare Erweiterungspunkte. Auf ein eingehendes Skript kann an jedem dieser Erweiterungspunkte verwiesen werden:

- RADIUS-Server
- Anbieter (des unmittelbaren Kunden)
- Client (einzelnes NAS)
- NAS-Anbieter-Behind-the-Proxy
- Client-Behind-the-Proxy
- Remote-Server (vom Typ RADIUS)
- Service

Auf ein Authentifizierungs- oder Autorisierungsskript kann an jedem dieser Erweiterungspunkte verwiesen werden:

- Gruppenauthentifizierung
- Benutzerauthentifizierung
- Gruppenautorisierung
- Benutzerautorisierung

Auf das ausgehende Skript kann an jedem dieser Erweiterungspunkte verwiesen werden:

- Service
- Client-Behind-the-Proxy
- NAS-Anbieter-Behind-the-Proxy
- Client (einzelnes NAS)
- NAS-Anbieter
- RADIUS-Server

Es ist wichtig, die Reihenfolge zu verstehen, in der Skripts von CPAR ausgeführt werden, da es mehrere Erweiterungspunkte gibt. In Tabelle 7-1 des Administratorhandbuchs finden Sie die Reihenfolge von 29 verfügbaren Scripting-/Erweiterungspunkten.

Ein internes Skript ist ein Skript, das direkt in der CPAR-CLI (aregcmd) konfiguriert wird. Es erfordert keine externen Dateien und viel Programmierkenntnisse. Ein externes Skript ist ein Skript, das in einer Datei im Betriebssystem (CENTOS oder RHEL) gespeichert und nur in der CPAR-CLI referenziert wird.

Konfigurieren

Internes Skript für ausgehenden Datenverkehr

In internen Skripts können Sie die folgenden Modifizierer verwenden:

1. +rsp: - fügt Antworten hinzu und weist diese zu

2. RSP: - entfernt das Attribut aus der Antwort

3. #rsp: - ersetzt Attribut durch neuen Wert

4. oben können für die Aufgaben (Request/Incoming Packet and Env, das Umgebungswörterbuch) verwendet werden. Beispiele +req: oder -env:

Fügen Sie unter /Radius/Scripts ein internes Skript hinzu. Konfigurieren Sie zwei zusätzliche AVPs für die Rückgabe mit dem Access-Accept-Paket: Filter-ID und anbieterspezifische (zum Beitritt zur Sprachdomäne).

```
--> ls -R

[ //localhost/Radius/Scripts/addattr ]
    Name = addattr
    Description =
    Language = internal
    Statements/
        1. +rsp:Filter-Id=PhoneACL
        2. +rsp:Cisco-AVPair=device-traffic-class=voice

--> ls -R

[ Services/local-users ]
    Name = local-users
    Description =
    Type = local
    IncomingScript~ =
    OutgoingScript~ = addattr
    OutagePolicy~ = RejectAll
    OutageScript~ =
    UserList = Default
    EnableDeviceAccess = True
    DefaultDeviceAccessAction~ = DenyAll
    DeviceAccessRules/
        1. switches
```

Test mit lokalem Radclient:

```
--> simple
```

```
p011
--> p011 send
p014
--> p014
Packet: code = Access-Accept, id = 18, length = 64, attributes =
        Filter-Id = PhoneACL
        Cisco-AVPair = device-traffic-class=voice
```

Spuren:

```
07/31/2019 10:31:26.254: P2363: Running Service local-users's OutgoingScript: addattr
07/31/2019 10:31:26.254: P2363: Internal Script for 1   +rsp:Filter-Id=PhoneACL : Filter-Id =
PhoneACL
07/31/2019 10:31:26.254: P2363: Setting value PhoneACL for attribute Filter-Id
07/31/2019 10:31:26.254: P2363: Trace of Response Dictionary
07/31/2019 10:31:26.254: P2363: Trace of Access-Request packet
07/31/2019 10:31:26.254: P2363:    identifier = 18
07/31/2019 10:31:26.254: P2363:    length = 30
07/31/2019 10:31:26.254: P2363:    respauth = fb:63:14:3f:c1:fb:ac:03:7d:16:29:61:ba:ef:13:4f
07/31/2019 10:31:26.254: P2363:    Filter-Id = PhoneACL
07/31/2019 10:31:26.254: P2363: Internal Script for 2   +rsp:Cisco-AVPair=device-traffic-
class=voice : Cisco-AVPair = device-traffic-class=voice
07/31/2019 10:31:26.254: P2363: Setting value device-traffic-class=voice for attribute Cisco-
AVPair
07/31/2019 10:31:26.254: P2363: Trace of Response Dictionary
07/31/2019 10:31:26.254: P2363: Trace of Access-Request packet
07/31/2019 10:31:26.254: P2363:    identifier = 18
07/31/2019 10:31:26.254: P2363:    length = 64
07/31/2019 10:31:26.254: P2363:    respauth = fb:63:14:3f:c1:fb:ac:03:7d:16:29:61:ba:ef:13:4f
07/31/2019 10:31:26.254: P2363:    Filter-Id = PhoneACL
07/31/2019 10:31:26.254: P2363:    Cisco-AVPair = device-traffic-class=voice
```

Internes Skript für eingehenden Datenverkehr

Erstellen Sie ein neues Skript, das alle Benutzernamen im Format user@domain in anonym ersetzt, und wenden Sie es als eingehenden Skript für den Dienst an, den Sie verwenden.

Konfigurieren:

```
--> cd /Radius/Scripts

--> add test

--> set language internal

--> cd Statements

--> add 1

--> cd 1

--> set statements "#req:User-Name=~(.*)(@[a-z]+.[a-z]+)~\anonymous"

--> ls -R
```

```
[ //localhost/Radius/Scripts/test ]
    Name = test
    Description =
    Language = internal
    Statements/
        1. #env:User-Name=~(.*)~anonymous

--> ls -R /Radius/Services/employee-service/

[ /Radius/Services/employee-service ]
    Name = employee-service
    Description =
    Type = local
    IncomingScript~ = test
    OutgoingScript~ =
    OutagePolicy~ = RejectAll
    OutageScript~ =
    UserList = default
    EnableDeviceAccess = FALSE
    DefaultDeviceAccessAction~ = DenyAll
```

Testen Sie mit radclient (Anfrage wird höchstwahrscheinlich abgelehnt, da der Benutzername in anonym geändert wird):

```
--> simple




p01e

--> p01e
Packet: code = Access-Request, id = 27, length = 72, attributes =
User-Name = <username>@cisco.com
User-Password = <password>
NAS-Identifier = localhost
NAS-Port = 7
--> p01e send
p020
--> p020
Packet: code = Access-Reject, id = 27, length = 35, attributes =
        Reply-Message = Access Denied
```

Nachverfolgung:

Vor Ausführung des Mitarbeiterservice werden drei Skripts aufgerufen. Zuerst ruft CPAR *CiscoIncomingScript auf*, dann ruft es *ParseServiceHints* auf, die an die Konfiguration des lokalhost-Client/Netzwerkgeräts angefügt ist. Es extrahiert den Benutzernamen aus dem Paket und legt ihn im Umgebungswörterbuch ab. Zweites Skript, *Test* wird aufgerufen und der Benutzername im Umgebungswörterbuch wird von <Benutzername> in anonym geändert.

localhost-Client:

```
[ //localhost/Radius/Clients/localhost ]
    Name = localhost
    Description =
```

```
    Protocol = radius
    IPAddress = 127.0.0.1
    SharedSecret = <encrypted>
    Type = NAS+Proxy
    Vendor = Cisco
    IncomingScript~ = ParseServiceHints
    OutgoingScript~ =
    EnableDynamicAuthorization = FALSE
    NetMask =
    EnableNotifications = FALSE
    EnforceTrafficThrottling = TRUE
```

Ablaufverfolgungsausgabe:

```
07/31/2019 11:38:53.522: P2855: PolicyEngine: [SelectPolicy] Successful
07/31/2019 11:38:53.522: P2855: Using Client: localhost
07/31/2019 11:38:53.522: P2855: Using Vendor: Cisco
07/31/2019 11:38:53.522: P2855: Running Vendor Cisco's IncomingScript: CiscoIncomingScript
07/31/2019 11:38:53.522: P2855: Running Client localhost IncomingScript: ParseServiceHints
07/31/2019 11:38:53.522: P2855: Rex: environ->get( "Request-Type" ) -> "Access-Request"
07/31/2019 11:38:53.522: P2855: Rex: environ->get( "Request-Type" ) -> "Access-Request"
07/31/2019 11:38:53.522: P2855: Rex: environ->get( "User-Name" ) -> "<username>"


07/31/2019 11:38:53.522: P2855: Authenticating and Authorizing with Service employee-service
07/31/2019 11:38:53.522: P2855: Running Service employee-service's IncomingScript: test
07/31/2019 11:38:53.522: P2855: Numbered attribute got for the radius / tacacs packet. ignoring
# User-Name
07/31/2019 11:38:53.523: P2855: Numbered attribute got for the radius / tacacs packet. ignoring
# User-Name
07/31/2019 11:38:53.523: P2855: Numbered attribute got for the radius / tacacs packet. ignoring
# User-Name
07/31/2019 11:38:53.523: P2855: Internal Script for 1   #env:User-Name=~(.*)~anonymous : User-
Name = anonymous
07/31/2019 11:38:53.523: P2855: Setting value anonymous for attribute User-Name
07/31/2019 11:38:53.523: P2855: Trace of Environment Dictionary
07/31/2019 11:38:53.523: P2855:            User-Name = anonymous
07/31/2019 11:38:53.523: P2855:            NAS-Name-And-IPAddress = localhost (127.0.0.1)
07/31/2019 11:38:53.523: P2855:            Authorization-Service = employee-service
07/31/2019 11:38:53.523: P2855:            Source-Port = 51169
07/31/2019 11:38:53.523: P2855:            Authentication-Service = employee-service
07/31/2019 11:38:53.523: P2855:            Trace-Level = 1000
07/31/2019 11:38:53.523: P2855:            Destination-Port = 1812
07/31/2019 11:38:53.523: P2855:            Destination-IP-Address = 127.0.0.1
07/31/2019 11:38:53.523: P2855:            Source-IP-Address = 127.0.0.1
07/31/2019 11:38:53.523: P2855:            Enforce-Traffic-Throttling = TRUE
07/31/2019 11:38:53.523: P2855:            Request-Type = Access-Request
07/31/2019 11:38:53.523: P2855:            Script-Level = 6
07/31/2019 11:38:53.523: P2855:            Provider-Identifier = Default
07/31/2019 11:38:53.523: P2855:            Request-Authenticator =
5f:62:5a:72:0f:7b:a2:2a:9c:06:ba:2e:bd:f4:e4:4b
07/31/2019 11:38:53.523: P2855:            Realm = cisco.com
07/31/2019 11:38:53.523: P2855: Getting User anonymous's UserRecord from UserList Default
07/31/2019 11:38:53.523: P2855: Failed to get User anonymous's UserRecord from UserList Default
07/31/2019 11:38:53.523: P2855: Running Vendor Cisco's OutgoingScript: CiscoOutgoingScript
07/31/2019 11:38:53.523: P2855: Trace of Access-Reject packet
07/31/2019 11:38:53.523: P2855:    identifier = 27
07/31/2019 11:38:53.523: P2855:    length = 35
07/31/2019 11:38:53.523: P2855:    respauth = d3:7d:b3:f6:05:47:2c:66:d9:c0:01:7d:67:d7:93:99
07/31/2019 11:38:53.523: P2855:    Reply-Message = Access Denied
07/31/2019 11:38:53.523: P2855: Sending response to 127.0.0.1
```

Externes Skript erstellen

Fügen Sie eine Datei *nadip.tcl* in das Verzeichnis "/opt/CSCOar/scripts/radius/tcl/" hinzu, und fügen Sie diesen Inhalt hinzu:

```
[root@piborowi-cpar80-16 tcl]# cat /opt/CSCOar/scripts/radius/tcl/nadip.tcl
proc UpdateNASIP {request response environ} {
$request trace 2 "TCL CUSTOM_SCRIPT Updating NAS IP ADDRESS"
$request trace 2 "Before put: " [ $request get NAS-IP-Address ]
$request put NAS-IP-Address 1.2.3.4
$request trace 2 "After put: " [ $request get NAS-IP-Address ]
}
```

Inhalt von *nadip.tcl* erklärt Zeile für Zeile:

Definition und Argumente des Verfahrens für Zeile 1. Anfrage, Antwort, Umgebung und drei verfügbare Wörterbücher, in denen Sie Sitzungs-/Paketdaten ändern können.

Zeile Nr. 2 Debug-Zeile für Skript, das als Ablaufverfolgungsebene 2 gedruckt werden soll.

Zeile Nr. 3 Inhalt des NAS-IP-Adresse-Attributs im Anforderungswörterbuch, bevor Sie diesen Wert festlegen.

Zeile 4 Legen Sie das Attribut "Nas-IP-Adresse" im Anforderungswörterbuch auf Wert 1.2.3.4 fest.

Zeile 5 Print NAS-IP-Address-Attribut erneut.

Konfigurieren Sie nach dem Erstellen und Speichern von Skripts im Betriebssystem den CPAR-Verweis auf das Skript. Als Sprache TCL festlegen, muss der Dateiname exakt den Dateinamen mit der Erweiterung sein (in diesem Fall ist es nadip.tcl). EntryPoint ist der Name der Prozedur in der Datei, die Sie als Skript ausführen möchten. Referenz erstellt CPAR Skript under Service (incomingScript) und Test mit Radclient.

Zeilen Nr. 2, Nr. 3 und Nr. 5 sind in der Spur zu sehen:

```
--> ls -R /Radius/scripts/nadipaddress/

[ /Radius/Scripts/nadipaddress ]
    Name = nadipaddress
    Description =
    Language = tcl             <<<<<<<<
    Filename = nadip.tcl       <<<<<<<<
    EntryPoint = UpdateNASIP   <<<<<<<<
    InitEntryPoint =
    InitEntryPointArgs =

--> ls -R /Radius/services/employee-service/

[ /Radius/Services/employee-service ]
    Name = employee-service
    Description =
    Type = local
    IncomingScript~ = nadipaddress    <<<<<<<<
    OutgoingScript~ =
    OutagePolicy~ = RejectAll
```

```
    OutageScript~ =
    UserList = default
    EnableDeviceAccess = FALSE
    DefaultDeviceAccessAction~ = DenyAll
```

Nachverfolgung:

```
07/31/2019 13:40:53.615: P3490: Running Service employee-service's IncomingScript: nadipaddress
07/31/2019 13:40:53.615: P3490: TCL CUSTOM_SCRIPT Updating NAS IP ADDRESS
07/31/2019 13:40:53.616: P3490:     Tcl: request trace 2 TCL CUSTOM_SCRIPT Updating NAS IP
ADDRESS -> OK
07/31/2019 13:40:53.616: P3490:     Tcl: request get NAS-IP-Address -> <empty>
07/31/2019 13:40:53.616: P3490: Before put:
07/31/2019 13:40:53.616: P3490:     Tcl: request trace 2 Before put:   -> OK
07/31/2019 13:40:53.616: P3490:     Tcl: request put NAS-IP-Address 1.2.3.4 -> OK
07/31/2019 13:40:53.616: P3490:     Tcl: request get NAS-IP-Address -> 1.2.3.4
07/31/2019 13:40:53.616: P3490: After put: 1.2.3.4
07/31/2019 13:40:53.616: P3490:     Tcl: request trace 2 After put:  1.2.3.4 -> OK
```