在CPAR 8.0上配置自定义脚本

目录

简介

本文档介绍如何使用脚本和扩展点自定义Cisco Prime Access Registrar(CPAR)8.0行为。

先决条件

要求

Cisco 建议您了解以下主题：

- CPAR 8.0管理

使用的组件

本文档中的信息基于以下软件和硬件版本：

- 安装在CentOS 6.5 64位上的CPAR 8.0

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

背景信息

CPAR可由内部和外部脚本修改。脚本可以用C/C++/Java/TCL编写。脚本可用于修改RADIUS、TACACS和DIAMETER数据包的处理。脚本可在扩展点的CPAR中引用。扩展点是显示在某些配置元素下并允许引用脚本的设置/属性。根据参[考指南](#),CPAR不对自定义脚本造成的任何数据丢失、损坏等负责。

以下是网络设备配置下两个扩展点的示例

```
[ //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
```

根据CPAR管理指南，有多个可用的扩展点。传入脚本可在以下每个扩展点引用：

- RADIUS 服务器
- 供应商（直接客户端）
- 客户端（单个NAS）
- NAS供应商代理后
- 客户端代理后
- 远程服务器（RADIUS类型）
- 服务

身份验证或授权脚本可在以下每个扩展点引用：

- 组身份验证
- 用户身份验证
- 组授权
- 用户授权

外发脚本可在以下每个扩展点引用：

- 服务
- 客户端代理后
- NAS供应商代理后
- 客户端（单个NAS）
- NAS供应商
- RADIUS 服务器

了解CPAR执行脚本的顺序至关重要，因为有多个扩展点。请参阅管理员指南的表7-1，查看29个可用脚本/扩展点的顺序。

内部脚本是直接在CPAR CLI(aregcmd)中配置的脚本。 它不需要任何外部文件和很多编程知识。外部脚本是存储在操作系统（CENTOS或RHEL）中的文件中，并刚在CPAR CLI中引用的脚本。

配置

外发流量的内部脚本

在内部脚本中，可以使用以下修饰符：

1.+rsp: — 添加和属性以响应

2.-rsp: — 从响应中删除属性

3.#rsp: — 用新值替换属性

4.以上内容可用于请求（请求/输入数据包和env，即环境字典）。 示例+req:或 — env:

在/Radius/Scripts下添加内部脚本。配置另外两个AVP，使用Access-Accept数据包返回：过滤器ID和供应商特定ID（加入语音域）。

```
--> 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
```

使用本地radclient进行测试：

```
--> simple
```

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

跟踪：

```
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
```

传入流量的内部脚本

创建一个新脚本，将user@domain格式的所有用户名替换为匿名，并将其作为您使用的服务的输入脚本应用。

配置:

```
--> 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
```

使用radclient进行测试（由于用户名更改为匿名，请求很可能被拒绝）：

```
--> 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
```

跟踪：

在执行员工服务之前，会调用三个脚本。首先，CPAR调*用CiscoIncomingScript*，然后调用*ParseServiceHints*，该Hints与本地主机客户端/网络设备配置连接。它从数据包中提取用户名并将其放入环境字典。第二个脚本，*调*用测试，并将环境词典中的用户名从<username>更改为匿名

localhost客户端：

```
[ //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
```

跟踪输出：

```
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
```

创建外部脚本

向/opt/CSCOar/scripts/radius/tcl/目*录中*添加文件nadip.tcl，并添加以下内容：

```
[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 ]
}
```

nadip.tcl*的内*容逐行说明：

行#1过程定义和参数。请求、响应、环境和三个可用字典，您可以在其中修改会话/数据包数据。

要作为跟踪级别2打印的脚本的#2行调试行。

在设#3此值之前，请求字典中NAS-IP-Address属性的行内容。

第#4行将请求字典中的Nas-IP-Address属性设置为值1.2.3.4。

第#5行再次打印NAS-IP-Address属性。

在操作系统中创建并保存脚本后，请配置对脚本的CPAR引用。将语言设置为TCL，文件名必须是带扩展名的确切文件名（本例中为nadip.tcl）。 EntryPoint是要作为脚本执行的文件中过程的名称。引用在服务(incomingScript)下创建的CPAR脚本，并使用radclient进行测试。

在跟踪#2中可以看到#3、#5行：

```
--> 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
```

跟踪：

```
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
```