Configurar scripts personalizados no CPAR 8.0

Contents

Introduction

Este documento descreve como personalizar o comportamento do Cisco Prime Access Registrar (CPAR) 8.0 com o uso de scripts e pontos de extensão.

Prerequisites

Requirements

A Cisco recomenda que você tenha conhecimento destes tópicos:

- administração de CPAR 8.0

Componentes Utilizados

As informações neste documento são baseadas nestas versões de software e hardware:

- CPAR 8.0 instalado no CentOS 6.5 64 bits

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Se a rede estiver ativa, certifique-se de que você entenda o impacto potencial de qualquer comando.

Informações de Apoio

O CPAR pode ser modificado por scripts internos e externos. Os scripts podem ser escritos em C/C++/Java/TCL. Os scripts podem ser usados para modificar o processamento de pacotes RADIUS, TACACS e DIAMETER. Os scripts podem ser consultados no CPAR em pontos de extensão. Pontos de extensão é uma configuração/atributo que aparece em alguns dos elementos de configuração e permite referenciar um script. De acordo com o guia de referência, o CPAR não é responsável por qualquer perda de dados, danos, etc. causados por scripts personalizados.

Aqui está um exemplo de dois pontos de extensão na configuração do dispositivo de rede

```
[ //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
```

De acordo com o guia de administração do CPAR, há vários pontos de extensão disponíveis. Um script de entrada pode ser referenciado em cada um destes pontos de extensão:

- servidor RADIUS
- Fornecedor (do cliente imediato)
- Cliente (NAS individual)
- NAS-Vendor-Bebehind-the-Proxy
- Cliente por trás do proxy
- Servidor remoto (do tipo RADIUS)
- Serviço

Um script de autenticação ou autorização pode ser referenciado em cada um destes pontos de extensão:

- Autenticação de grupo
- Autenticação de usuário
- Autorização do grupo
- Autorização do usuário

O script de saída pode ser referenciado em cada um destes pontos de extensão:

- Serviço
- Cliente por trás do proxy
- NAS-Vendor-Bebehind-the-Proxy
- Cliente (NAS individual)
- Fornecedor de NAS
- servidor RADIUS

Éfundamental entender a ordem na qual os scripts são executados pelo CPAR, pois há vários pontos de extensão. Consulte a tabela 7-1 do guia do administrador para ver a ordem de 29 pontos de script/extensão disponíveis.

Um script interno é um que é configurado diretamente na CLI do CPAR (aregcmd). Não exige arquivos externos nem muito conhecimento de programação. Um script externo é um que é armazenado em um arquivo no sistema operacional (CENTOS ou RHEL) e é mencionado apenas na CLI do CPAR.

Configurar

Script Interno Para Tráfego De Saída

Em scripts internos, você pode usar estes modificadores:

1. +rsp: - adiciona e atribui à resposta

2. -rsp: - remove o atributo da resposta

3. #rsp: - substitui atributo por novo valor

4. acima pode ser usado para req (request/incoming packet and env, que é o dicionário de ambiente). Exemplos +solicitação: ou -env:

Adicione um Script interno em /Radius/Scripts. Configure dois AVP adicionais a serem retornados com o pacote Access-Accept: Filter-Id e Vendor-Specific one (para ingressar no domínio de voz).

```
--> 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
```

Teste com o uso de radclient local:

```
--> simple




p011
--> p011 send
p014
```

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

Rastreamentos:

```
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
```

Script Interno Para Tráfego De Entrada

Crie um novo script que substitua todos os nomes de usuário no formato user@domain para anônimo e o aplique como script de entrada para o serviço que você usa.

Configurar:

```
--> 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
```

Teste com radclient (a solicitação é provavelmente rejeitada porque o nome de usuário é alterado para anônimo):

```
--> 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
```

Rastreamento:

Antes de o serviço do funcionário ser executado, três scripts são chamados. Primeiro, o CPAR chama *CiscoIncomingScript* e, em seguida, chama *ParseServiceHands*, que está conectado à configuração do dispositivo cliente/rede localhost. Ele extrai o nome de usuário do pacote e o coloca no dicionário do ambiente. Segundo script, *teste* é chamado e o nome de usuário no dicionário de ambiente é alterado de **<**nome de usuário**>** para anônimo

cliente de host local:

```
[ //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
```

Rastrear saída:

```
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
```

Criar Script Externo

Adicione um arquivo *nadip.tcl* ao diretório /opt/CSCOar/scripts/radius/tcl/ e adicione este conteúdo:

```
[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 ]
}
```

Conteúdo do *nadip.tcl* explicado linha por linha:

Linha 1 Definição e argumentos do procedimento. Solicitação, resposta, ambiente e três dicionários disponíveis nos quais você pode modificar dados de sessão/pacote.

Linha 2 Depurar linha para script a ser impresso como nível de rastreamento 2.

Linha 3 Conteúdo do atributo NAS-IP-Address no dicionário de solicitações antes de definir esse valor.

Linha 4 Defina o atributo Nas-IP-Address no dicionário de solicitação para o valor 1.2.3.4.

Linha nº 5 Imprima novamente o atributo NAS-IP-Address.

Quando o script for criado e salvo no sistema operacional, configure a referência CPAR para o script. Defina o idioma como TCL, o nome do arquivo deve ser o nome exato do arquivo com a extensão (nesse caso, é nadip.tcl). EntryPoint é o nome do procedimento no arquivo que você deseja executar como um script. Referência criada script CPAR em serviço (incomingScript) e teste com radclient.

As linhas 2, 3 e 5 podem ser observadas no rastreamento:

```
--> 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
```

Rastreamento:

```
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
```