



## CHAPTER 26

# Using the REX Accounting Script

---

This chapter describes how to use the REX Accounting Script (RAS). The RAS writes RADIUS Accounting requests to a local, flat file and is included as an option for Cisco Prime Access Registrar (Cisco Prime AR). It is designed to be attached to a Cisco Prime AR IncomingScript or OutgoingScript point. When used in conjunction with the Cisco Prime AR built-in proxy support, the server will concurrently store a local copy of an Accounting request and proxy another copy to another RADIUS server.



### Note

Unless you require log rotation at an exact time or when the accounting log reaches a specific file size, Cisco recommends that you use service grouping to log and proxy accounting packets.

---

RAS can be attached to more than one Cisco Prime AR extension point. For example, in a dial-up resale scenario, you might configure Cisco Prime AR to proxy Accounting requests to many different Remote Servers (by realm). For some subset of those, you might want to keep a local copy of the Accounting requests. In this case, RAS could be installed as the IncomingScript on just the Services for which a local copy is desired.



### Note

Also included is the **DropAcctOnOff** Script. This script causes Cisco Prime AR to drop all Accounting-Requests with an **Acct-Status-Type** of **Accounting-On** or **Accounting-Off**.

---

This chapter contains the following sections:

- [Building and Installing the REX Accounting Script](#)
- [Configuring the Rex Accounting Script](#)
- [Specifying REX Accounting Script Options](#)

## Building and Installing the REX Accounting Script

The RAS writes RADIUS Accounting requests to a local, flat file and is included as an option for Cisco Prime Access Registrar (Cisco Prime AR). It is designed to be attached to a Cisco Prime AR IncomingScript or OutgoingScript point.

**Building and Installing the REX Accounting Script**

To build and install RAS:

- 
- Step 1** Change directory to `$INSTALL/examples/rexacctscript`.
- Step 2** Modify the **Makefile** to ensure the `AR_INSTALL_DIR` variable points to the directory where the Cisco Prime AR software was installed, and then choose a compiler (`gcc` or `SUNPro CC`).
- Step 3** From the command line prompt, enter:
- ```
host% make
```
- Step 4** Log in as user **root**.
- Step 5** From the command line prompt, enter:
- ```
host# make install
```
- 

## Configuring the Rex Accounting Script

RAS can be attached to more than one Cisco Prime AR extension point. For example, in a dial-up resale scenario, you might configure Cisco Prime AR to proxy Accounting requests to many different Remote Servers (by realm).

**Configuring the Rex Accounting Script**

To configure RAS:

- 
- Step 1** Start the Cisco Prime AR **aregcmd** configuration utility and login:
- ```
> $INSTALL/usrbin/aregcmd -C localhost -N admin -P aicuser
Access Registrar Configuration Utility Version 1.3
Copyright (C) 1995-2008 by Cisco Systems, Inc. All rights reserved.
Logging in to localhost
[ //localhost ]
    LicenseKey = xxxx-xxxx-xxxx-xxxx
    Radius/
    Administrators/
Server 'Radius' is Running, its health is 10 out of 10
-->
```
- Step 2** Using **aregcmd**, create a new Cisco Prime AR Script object:
- ```
--> cd /Radius/Scripts
[ //localhost/Radius/Scripts ]
    Entries 1 to 20 from 39 total entries
    Current filter: <all>
    ACMEOutgoingScript/
    AscendIncomingScript/
<... other output deleted...>
--> add LocalAccounting
Added LocalAccounting
```

- Step 3** Using **aregcmd**, fill in the details of the new Cisco Prime AR Script object. See [Chapter 4, “Cisco Prime Access Registrar Server Objects,”](#) for more details.

--> **cd LocalAccounting**

```
[ //localhost/Radius/Scripts/LocalAccounting ]
    Name = LocalAccounting
    Description =
    Language =
    Filename =
    EntryPoint =
    InitEntryPoint =
    InitEntryPointArgs =
```

--> **set Desc “Log Accounting requests to local file”**

```
Set Description "Log Accounting requests to local file"
```

--> **set lang REX**

```
Set Language REX
```

--> **set filename libRexAcctScript.so**

```
Set Filename libRexAcctScript.so
```

--> **set entry RexAccountingScript**

```
Set EntryPoint RexAccountingScript
```

--> **set initentrypoint InitRexAccountingScript**

```
Set InitEntryPoint InitRexAccountingScript
```

--> **set initentrypointargs “-f Accounting -t 1:15”**

```
Set InitEntryPointArgs "-f Accounting -t 1:15"
```

--> **ls**

```
[ //localhost/Radius/Scripts/LocalAccounting ]
    Name = LocalAccounting
    Description = "Log Accounting requests to local file"
    Language = REX
    Filename = libRexAcctScript.so
    EntryPoint = RexAccountingScript
    InitEntryPoint = InitRexAccountingScript
    InitEntryPointArgs = "-f Accounting -t 1:15"
```

-->

- Step 4** Using **aregcmd**, attach the new Cisco Prime AR Script object to the appropriate Cisco Prime AR Scripting point. See [Chapter 4, “Cisco Prime Access Registrar Server Objects,”](#) for more details.

--> **set /radius/IncomingScript LocalAccounting**

```
Set /Radius/IncomingScript LocalAccounting
```

- Step 5** Using **aregcmd**, save the configuration modifications:

--> **save**

```
Validating //localhost...
```

```
Saving //localhost...
```

- Step 6** Using **aregcmd**, reload the server:

```
--> reload
```

```
Reloading Server 'Radius'...
```

```
Server 'Radius' is Running, its health is 10 out of 10
```

## Specifying REX Accounting Script Options

The REX Accounting Script supports the options shown in [Table 26-1](#).

**Table 26-1** REX Accounting Script Supported Options

Option	Description
<b>-f</b> <filename>	Required. Specify the name of the output file.
<b>-t</b> <HH:MM[:SS]>	Specify a time of day to roll the output file. Note, this is time on the 24-hour clock, for example, 00:05 = 12:05am, 13:30 = 1:30pm. This option can not be used with the <b>-i</b> option.
<b>-i</b> <seconds>	Specify the number of seconds between rolling the output file, beginning at start-up. This option can not be used with the <b>-t</b> option.
<b>-s</b> <size>[klmgl]	Specify the maximum size for an output file. When the file reaches this size, it will be rolled.  When specifying the <size> option, a <unit> can be included. When a <unit> is not included, the <size> is in bytes. Note, do not use a space character between the <size> and <unit> options.  <unit> can be either: k = 1K, m = 1Meg, g = 1Gig.
<b>-g</b>	Use GMT when writing the date/time in the Accounting output file for each record (default is local time).
<b>-G</b>	Use GMT when naming rolled output files (default is local time).
<b>-A</b>	Process all packets, not just Accounting-Requests.
<b>-I</b>	Ignore errors when processing packets, always return successfully.
<b>-a</b> <buffer-count>	Pre-allocate this many Accounting buffers to improve performance.
<b>-T</b> <trace-level>	Set the trace level. This trace info appears in the output file (as its written by the background thread which no longer has a packet to use for logging or tracing.)
<b>-O</b> <script-description>	Call another REX extension before calling the <b>RexAcctScript</b> .
<b>-o</b> <script-description>	Call another REX extension after calling the <b>RexAcctScript</b> .

## Example Script Object

This is an example of what a Cisco Prime AR Script object using RAS might look like when viewed in the Cisco Prime AR configuration utility, **aregcmd**:

```
[ //localhost/Radius/Scripts/REX-Accounting-Script ]
  Name = REX-Accounting-Script
  Description =
  Language = REX
  Filename = librexacctscript.so
  EntryPoint = RexAccountingScript
  InitEntryPoint = InitRexAccountingScript
  InitEntryPointArgs = "-f Accounting -t 16:20 -s 100k -o
    libRexAcctScript.so:DropAcctOnOff"
```

This example causes RAS to write to a file called **Accounting.log** (in the **logs** directory of the installation tree). The file rolls every day at 4:20pm (local time), as well as whenever it grows larger than 100k in size. RAS also runs the **DropAcctOnOff** script against every packet, after it has processed the packet.

