

Configuring Cisco Secure UNIX and Secure ID (SDI Client)

Document ID: 5610

Contents

Introduction

Prerequisites

- Requirements
- Components Used
- Conventions

Installing an SDI Client (Secure ID) on a Cisco Secure UNIX Machine

Initial Testing of the Secure ID and CSUnix

Secure ID and CSUnix: TACACS+ Profile

- How the Profile Works
- CSUnix TACACS+ Password Combinations that do not Work
- Debugging CSUnix TACACS+ SDI Sample Profiles

CSUnix RADIUS

- Login Authentication with CSUnix and RADIUS
- PPP and PAP Authentication with CSUnix and RADIUS
- Dial-up Networking PPP Connection and PAP

Debug and Verification Tips

- Cisco Secure RADIUS, PPP, and PAP
- Secure ID and CSUnix

Related Information

Introduction

To implement the configuration in this document, you need any Cisco Secure version that supports Security Dynamics Incorporated (SDI)'s Secure ID.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

This document is not restricted to specific software and hardware versions.

Conventions

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Installing an SDI Client (Secure ID) on a Cisco Secure UNIX Machine

Note: Secure ID is usually installed before Cisco Secure UNIX (CSUnix) has been installed. These instructions describe how to install the SDI client after CSUnix has been installed.

1. On the SDI server, run **sdadmin**. Tell the SDI server that the CSUnix machine is a client and specify that the SDI users in question are activated on the CSUnix client.
2. Use the **nslookup #.#.#.#** or **nslookup <hostname>** command to make sure the CSUnix client and the SDI server can do forward and reverse lookup of each other.
3. Copy the SDI server's /etc/sdace.txt file to CSUnix client /etc/sdace.txt file.
4. Copy the SDI server's sdconf.rec file to the CSUnix client; this file may reside anywhere on the CSUnix client. However, if it is placed in the same directory structure on the CSUnix client as it was on the SDI server, sdace.txt does not have to be modified.
5. Either /etc/sdace.txt or VAR_ANCE must point to the path where the sdconf.rec file is located. To verify this, run cat /etc/sdace.txt, or check the output of env to be sure that VAR_ANCE is defined in the root's profile as the root starts.
6. Back up the CSUnix client's CSU.cfg, then modify the AUTHEN config_external_authen_symbols section with these lines:

```
AUTHEN config_external_authen_symbols = {
  {
    "./libskey.so",
    "skey"
  }
  ,
  {
    "./libsdi.so",
    "sdi"
  }
  ,
  {
    "./libpap.so",
    "pap"
  }
  ,
  {
    "./libchap.so",
    "chap"
  }
}
```

Note: A "," is required before and after these lines if preceded or followed by another option "AUTHEN config_external_authen_symbols" section in the CSU.cfg file. The "," is *not* required when these lines appear as the last lines of the "AUTHEN config_external_authen_symbols" section of the CSU.cfg file.

7. Recycle CSUnix by the execution of **K80CiscoSecure** and **S80CiscoSecure**.
8. If \$BASE/utils/psg shows that the Cisco Secure AAA Server Process process was active before the CSU.cfg file was modified but not afterwards, then errors were made in the revision of the CSU.cfg file. Restore the original CSU.cfg file and try to make the changes outlined in step 6 again.

Initial Testing of the Secure ID and CSUnix

To test Secure ID and CSUnix, perform these steps:

1. Make sure that a non-SDI user can Telnet to the router and be authenticated with CSUnix. If this does not work, SDI will not work.
2. Test basic SDI authentication in the router and run this command:

```
aaa new-model
aaa authentication login default tacacs+ none
```

Note: This assumes that the **tacacs-server** commands are already active in the router.

3. Add an SDI user from the CSUnix command line to enter this command

```
$BASE/CLI/AddProfile -p 9900 -u sdi_user -pw sdi
```

4. Try to authenticate as an user. . If that user works, SDI is operational, and you can add additional information to the user profiles.
5. SDI users can be tested with the unknown_user profile in CSUnix. (Users do not have to be explicitly listed in CSUnix if they all are passed off to SDI and all have the same profile.) If there is an unknown user profile already exist, delete it with the help of this command:

```
$BASE/CLI/DeleteProfile -p 9900 -u unknown_user
```

6. Use this command to add another unknown user profile:

```
$BASE/CLI/AddProfile -p 9900 -u unknown_user -pw sdi
```

This command passes off all unknown users to SDI.

Secure ID and CSUnix: TACACS+ Profile

1. Perform an initial test without SDI. If this user profile does not work without an SDI password for login authentication, Challenge Handshake Authentication Protocol (CHAP), and Password Authentication Protocol (PAP), it will not work with an SDI password:

```
# ./ViewProfile -p 9900 -u cse
User Profile Information
user = cse{
password = chap "chappwd"
password = pap "pappwd"
password = clear,"clearpwd"
default service=permit
service=shell {
}
service=ppp {
protocol=lcp {
}
}
protocol=ip {
}
}
}
```

2. Once the profile works, add "sdi" to the profile in place of "clear" as shown in this example:

```
# ./ViewProfile -p 9900 -u cse
User Profile Information
user = cse{
password = chap "chappwd"
password = pap "pappwd"
password = sdi
default service=permit
service=shell {
}
service=ppp {
protocol=lcp {
}
}
protocol=ip {
}
}
}
```

How the Profile Works

This profile allows the user to log in with these combinations:

- Telnet to the router and use SDI. (This assumes that the **aaa authentication login default tacacs+** command has been executed on the router.)
- Dial-up networking PPP connection and PAP. (This assumes that the **aaa authentication ppp default if-needed tacacs** and **ppp authen pap** commands have been executed on the router).

Note: On the PC, in Dial-Up Networking, make sure "Accept any authentication including clear text" is checked. Before dialing, enter one of these username/password combinations in the terminal window:

```
username: cse*code+card
password: pap (must agree with profile)
```

```
username: cse
password: code+card
```

- Dial-up networking PPP connection and CHAP. (This assumes that the **aaa authentication ppp default if-needed tacacs** and **ppp authen chap** commands have been executed on the router).

Note: On the PC, in Dial-Up Networking, either "Accept any authentication including clear text" or "Accept only encrypted authentication" must be checked. Before dialing, enter this username and password in the terminal window:

```
username: cse*code+card
password: chap (must agree with profile)
```

CSUnix TACACS+ Password Combinations that do not Work

These combinations produce these CSUnix debug errors:

- CHAP and no "cleartext" password in the password field. The user enters code+card instead of the "cleartext" password. RFC 1994 on CHAP requires clear text password storage.

```
username: cse
password: code+card
```

```
CiscoSecure INFO - User cse, No tokencard password received
CiscoSecure NOTICE - Authentication - Incorrect password;
```

- CHAP and a bad CHAP password.

```
username: cse*code+card
password: wrong chap password
```

(The user passes off to SDI, and SDI passes the user, but CSUnix fails the user because the CHAP password is bad.)

```
CiscoSecure INFO - The character * was found in username:
  username=cse,passcode=1234755962
CiscoSecure INFO - sdi_challenge: rtn 1, state=GET_PASSCODE, user=cse
CiscoSecure INFO - sdi_verify: cse authenticated by ACE Srvr
CiscoSecure INFO - sdi: cse free external_data memory,state=GET_PASSCODE
CiscoSecure INFO - sdi_verify: rtn 1
CiscoSecure NOTICE - Authentication - Incorrect password;
```

- PAP and a bad PAP password.

```
username: cse*code+card
password: wrong pap password
```

(The user passes off to SDI, and SDI passes the user, but CSUnix fails the user because the CHAP password is bad.)

```

CiscoSecure INFO - 52 User Profiles and 8 Group Profiles loaded into Cache.
CiscoSecure INFO - The character * was found in username:
    username=cse,passcode=1234651500
CiscoSecure INFO - sdi_challenge: rtn 1, state=GET_PASSCODE, user=cse
CiscoSecure INFO - sdi_verify: cse authenticated by ACE Srvr
CiscoSecure INFO - sdi: cse free external_data memory,state=GET_PASSCODE
CiscoSecure INFO - sdi_verify: rtn 1
CiscoSecure NOTICE - Authentication - Incorrect password;

```

Debugging CSUnix TACACS+ SDI Sample Profiles

- The user needs to do CHAP and login authentication; PAP fails.

```

# ./ViewProfile -p 9900 -u cse
User Profile Information
user = cse{
password = chap "*****"
password = sdi
default service=permit
service=shell {
}
service=ppp {
protocol=lcp {
}
}
protocol=ip {
}
}

```

- The user needs to do PAP and login authentication; CHAP fails.

```

# ./ViewProfile -p 9900 -u cse
User Profile Information
user = cse{
member = admin
password = pap "*****"
password = sdi
default service=permit
service=shell {
}
}
service=ppp {
protocol=lcp {
}
}
protocol=ip {
}
}
}

```

CSUnix RADIUS

These sections contain CSUnix RADIUS procedures.

Login Authentication with CSUnix and RADIUS

Perform these steps to test authentication:

1. Perform an initial test without SDI. If this user profile does not work without an SDI password for login authentication, it will not work with an SDI password:

```

# ./ViewProfile -p 9900 -u cse
User Profile Information
user = cse{
radius=Cisco {

```

```

check_items= {
2="whatever"
}
reply_attributes= {
6=6
}
}
}

```

2. Once this profile works, replace "whatever" with "sdi" as shown in this example:

```

# ./ViewProfile -p 9900 -u cse
User Profile Information
user = cse{
radius=Cisco {
check_items= {
2=sdi
}
reply_attributes= {
6=6

}
}
}

```

PPP and PAP Authentication with CSUnix and RADIUS

Perform these steps to test authentication:

Note: PPP CHAP authentication with CSUnix and RADIUS is not supported.

1. Perform an initial test without SDI. If this user profile does not work without an SDI password for PPP/PAP authentication and "async mode dedicated," it will not work with an SDI password:

```

# ./ViewProfile -p 9900 -u cse

user = cse {
password = pap "pappass"
radius=Cisco {
check_items = {
}
reply_attributes= {
6=2
7=1
}
}
}

```

2. Once the above profile works, add **password = sdi** to the profile and add attribute **200=1** as shown in this example (this sets Cisco_Token_Immediate to yes.):

```

# ./ViewProfile -p 9900 -u cse
user = cse {
password = pap "pappass"
password = sdi
radius=Cisco {
check_items = {
200=1
}
reply_attributes= {
6=2
7=1
}
}
}
}

```

3. In the "**Advanced GUI**, server section," **make sure "Enable Token Caching"** is set. This can be confirmed from the command line interface (CLI) with:

```
$BASE/CLI/ViewProfile -p 9900 -u SERVER.#.#.#.#  
  
!--- Where #.#.#.# is the IP address of the CSUnix server.  
  
TokenCachingEnabled="yes"
```

Dial-up Networking PPP Connection and PAP

It is assumed that **aaa authentication ppp default if-needed tacacs** and **PPP authen PAP** commands have been executed on the router. Enter this username and password in the terminal window before you dial.:

```
username: cse  
password: code+card
```

Note: On the PC, in Dial-Up Networking, make sure "Accept any authentication including clear text" is checked.

Debug and Verification Tips

These sections contain tips for debug and verification tips.

Cisco Secure RADIUS, PPP, and PAP

This is an example of a good debug:

```
CiscoSecure DEBUG - RADIUS ; Outgoing Accept Packet id=133 (10.31.1.6)  
  User-Service-Type = Framed-User  
  Framed-Protocol = PPP  
CiscoSecure DEBUG - RADIUS ; Request from host alf0106 nas (10.31.1.6)  
  code=1 id=134 length=73  
CiscoSecure DEBUG - RADIUS ; Incoming Packet id=134 (10.31.1.6)  
  Client-Id = 10.31.1.6  
  Client-Port-Id = 1  
  NAS-Port-Type = Async  
  User-Name = "cse"  
  Password = "?\235\306"  
  User-Service-Type = Framed-User  
  Framed-Protocol = PPP  
CiscoSecure DEBUG - RADIUS ; Authenticate (10.31.1.6)  
CiscoSecure DEBUG - RADIUS ; checkList: ASCEND_TOKEN_IMMEDIATE = 1  
CiscoSecure DEBUG - RADIUS ; User PASSWORD type is Special  
CiscoSecure DEBUG - RADIUS ; authPapPwd (10.31.1.6)  
CiscoSecure INFO - sdi_challenge: rtn 1, state=GET_PASSCODE, user=cse  
CiscoSecure DEBUG - profile_valid_tcaching FALSE ending.  
CiscoSecure DEBUG - Token Caching. IGNORE.  
CiscoSecure INFO - sdi_verify: cse authenticated by ACE Srvr  
CiscoSecure INFO - sdi: cse free external_data memory,state=GET_PASSCODE  
CiscoSecure INFO - sdi_verify: rtn 1  
CiscoSecure DEBUG - RADIUS ; Sending Ack of id 134 to alf0106 (10.31.1.6)
```

Secure ID and CSUnix

The debug is stored in the file specified in /etc/syslog.conf for local0.debug.

No users can authenticate – SDI or otherwise:

After you add the Secure ID, make sure that no errors were made when you modify the CSU.cfg file. Fix the CSU.cfg file or revert to the backup CSU.cfg file.

This is an example of a good debug:

```
Dec 13 11:24:22 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_challenge: rtn 1, state=GET_PASSCODE, user=cse
Dec 13 11:24:22 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_challenge: rtn 1, state=GET_PASSCODE, user=cse
Dec 13 11:24:31 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_verify: cse authenticated by ACE Srvr
Dec 13 11:24:31 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_verify: cse authenticated by ACE Srvr
Dec 13 11:24:31 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi: cse free external_data memory,state=GET_PASSCODE
Dec 13 11:24:31 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi: cse free external_data memory,state=GET_PASSCODE
Dec 13 11:24:31 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_verify: rtn 1
Dec 13 11:24:31 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_verify: rtn 1
```

This is an example of a bad debug:

CSUnix finds the user profile and sends it to the SDI server, but the SDI server fails the user because the passcode is bad.

```
Dec 13 11:26:22 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_challenge: rtn 1, state=GET_PASSCODE, user=cse
Dec 13 11:26:22 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_challenge: rtn 1, state=GET_PASSCODE, user=cse
Dec 13 11:26:26 rtp-evergreen.rtp.cisco.com CiscoSecure:
  WARNING - sdi_verify: cse denied access by ACE Srvr
Dec 13 11:26:26 rtp-evergreen.rtp.cisco.com CiscoSecure:
  WARNING - sdi_verify: cse denied access by ACE Srvr
Dec 13 11:26:26 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi: cse free external_data memory,state=GET_PASSCODE
Dec 13 11:26:26 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi: cse free external_data memory,state=GET_PASSCODE
Dec 13 11:26:26 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_verify: rtn 0
Dec 13 11:26:26 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi_verify: rtn 0
Dec 13 11:26:26 rtp-evergreen.rtp.cisco.com CiscoSecure:
  NOTICE - Authentication - Incorrect password;
Dec 13 11:26:26 rtp-evergreen.rtp.cisco.com CiscoSecure:
  NOTICE - Authentication - Incorrect password;
```

This is an example show that the Ace server is down:

Enter **./aceserver stop** on the SDI server. The user does not get the "Enter PASSCODE" message.

```
Dec 13 11:33:42 rtp-evergreen.rtp.cisco.com CiscoSecure:
  ERROR - sdi_challenge error: sd_init failed cli/srvr comm init (cse)
Dec 13 11:33:42 rtp-evergreen.rtp.cisco.com CiscoSecure:
  ERROR - sdi_challenge error: sd_init failed cli/srvr comm init (cse)
Dec 13 11:33:42 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi: cse free external_data memory,state=RESET
Dec 13 11:33:42 rtp-evergreen.rtp.cisco.com CiscoSecure:
  INFO - sdi: cse free external_data memory,state=RESET
```

Related Information

- [Documentation for Cisco Secure ACS for UNIX](#)
 - [Cisco Secure ACS for UNIX Support Page](#)
 - [Field Notices for Cisco Secure ACS for UNIX](#)
 - [Technical Support – Cisco Systems](#)
-

[Contacts & Feedback](#) | [Help](#) | [Site Map](#)

© 2009 – 2010 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)

Updated: May 14, 2009

Document ID: 5610
