

Configuring Authentication

Authentication identifies users before they are allowed access to the network and network services. Basically, the Cisco IOS software implementation of authentication is divided into two main categories:

- AAA Authentication Methods
- Non-AAA Authentication Methods

Authentication, for the most part, is implemented through the AAA security services. We recommend that, whenever possible, AAA be used to implement authentication.

This chapter describes both AAA and non-AAA authentication methods. For configuration examples, refer to the “Authentication Configuration Examples” section at the end of this chapter. For a complete description of the commands used in this chapter, refer to the “AAA Authentication Commands” chapter of the *Security Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

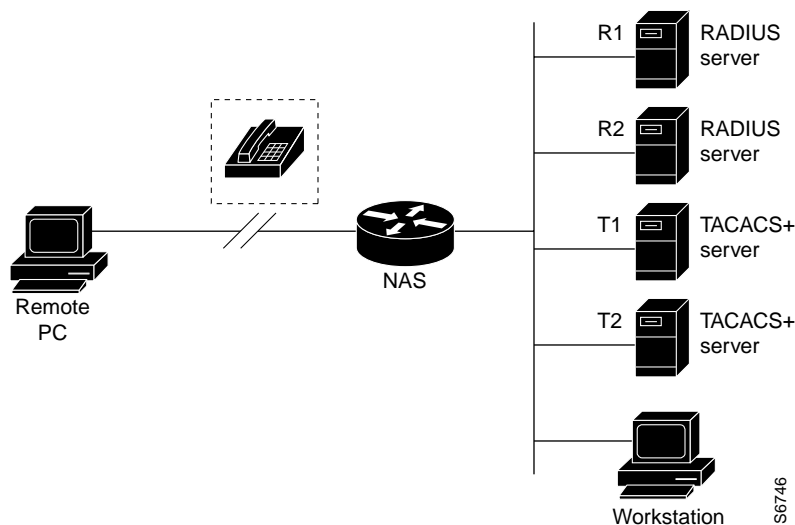
AAA Authentication Method Lists

To configure AAA authentication, first define a named list of authentication methods, and then apply that list to various interfaces. The method list defines the types of authentication to be performed and the sequence in which they will be performed; it must be applied to a specific interface before any of the defined authentication methods will be performed. The only exception is the default method list (which, by coincidence, is named “default”). The default method list is automatically applied to all interfaces except those that have a named method list explicitly defined. A defined method list overrides the default method list.

A method list is simply a list describing the authentication methods to be queried, in sequence, to authenticate a user. Method lists enable you to designate one or more security protocols to be used for authentication, thus ensuring a backup system for authentication in case the initial method fails. Cisco IOS software uses the first method listed to authenticate users; if that method fails to respond, the Cisco IOS software selects the next authentication method listed in the method list. This process continues until there is successful communication with a listed authentication method, or all methods defined are exhausted.

It is important to note that the Cisco IOS software attempts authentication with the next listed authentication method only when there is no response from the previous method. If authentication fails at any point in this cycle—meaning that the security server or local username database responds by denying the user access—the authentication process stops and no other authentication methods are attempted.

Figure 3 Typical AAA Network Configuration



Method List Examples

Figure 3 shows a typical AAA network configuration that includes four security servers: R1 and R2 are RADIUS servers and T1 and T2 are TACACS+ servers. Suppose the system administrator has decided on a security solution where all interfaces will use the same authentication methods to authenticate PPP connections: R1 is contacted first for authentication information, then if there is no response, R2 is contacted. If R2 fails to respond, T1 is contacted; if T1 fails to respond, T2 is contacted. If all designated servers fail to respond, authentication falls over to the local username database on the access server itself. To implement this, the system administrator would create a default method list by entering the following command:

```
aaa authentication ppp default radius tacacs+ local
```

In this example, “default” is the name of the method list. The protocols included in this method list are listed after the name, in the order they are to be queried. The default list is automatically applied to all interfaces.

Suppose the system administrator wanted to apply this method list only to a particular interface or set of interfaces. In this case, the system administrator would create a named method list and then apply this named list to the applicable interfaces. The following example shows how the system administrator would implement the previous scenario if that authentication method were to be applied only to interface 3:

```
aaa authentication ppp apple radius tacacs+ local
interface async 3
ppp authentication chap apple
```

In this example, “apple” is the name of the method list, and the protocols included in this method list are listed after the name in the order in which they are to be performed. After the method list has been created, it is applied to the appropriate interface. Please note that the method list name in both the **aaa authentication** command and the **ppp authentication** commands must match.

When a remote user attempts to dial in to the network, the network access server first queries R1 for authentication information. If R1 authenticates the user, it issues a PASS response to the network access server and the user is allowed to access the network. If R1 returns a FAIL response, the user is denied access and the session is terminated. If R1 fails to respond, then the network access server

processes that as an ERROR and queries R2 for authentication information. This pattern would continue through the remaining designated methods until the user is either authenticated, rejected, or the session terminated.

It is important to remember that a FAIL response is significantly different from an ERROR. A FAIL means that the user has not met the criteria contained in the applicable authentication database to be successfully authenticated. Authentication ends with a FAIL response. An ERROR means that the security server has failed to respond to an authentication query. Because of this, no authentication has been attempted. Only when an ERROR is detected will AAA select the next authentication method defined in the authentication method list.

AAA Authentication General Configuration Procedure

To configure AAA authentication, no matter what method of authentication you select, you need to perform the following tasks:

- 1 Enable AAA by using the **aaa new-model** global configuration command. For more information about configuring AAA, refer to the “AAA Overview” chapter.
- 2 Configure security protocol parameters, such as RADIUS, TACACS+, or Kerberos, if you are using a security server. For more information about RADIUS, refer to the “Configuring RADIUS” chapter. For more information about TACACS+, refer to the “Configuring TACACS+” chapter. For more information about Kerberos, refer to the “Configuring Kerberos” chapter.
- 3 Define the method lists for authentication by using the **aaa authentication** command.
- 4 Apply the method lists to a particular interface or line, if required.

AAA Authentication Methods

This section discusses the following AAA authentication methods:

- Configure Login Authentication Using AAA
- Configure PPP Authentication Using AAA
- Configure ARA Authentication Using AAA
- Configure NASI Authentication Using AAA
- Enable Password Protection at the Privileged Level
- Enable an Authentication Override
- Enable Double Authentication

Note AAA features are not available for use until you enable AAA globally by issuing the **aaa new-model** command. For more information about enabling AAA, refer to the “AAA Overview” chapter.

Configure Login Authentication Using AAA

The AAA security services facilitate a variety of login authentication methods. Use the **aaa authentication login** command to enable AAA authentication no matter which of the supported login authentication methods you decide to use. With the **aaa authentication login** command, you create one or more lists of authentication methods that are tried at login. These lists are applied using the **login authentication** line configuration command.

To configure login authentication by using AAA, perform the following tasks, beginning in global configuration mode:

Task	Command
Enable AAA globally.	aaa new-model
Create a local authentication list.	aaa authentication login { default <i>list-name</i> } <i>method1</i> [<i>method2...</i>]
Enter line configuration mode for the lines to which you want to apply the authentication list.	line [aux console tty vtty] <i>line-number</i> [<i>ending-line-number</i>]
Apply the authentication list to a line or set of lines.	login authentication { default <i>list-name</i> }

The keyword *list-name* is any character string used to name the list you are creating. The keyword *method* refers to the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

For example, to specify that authentication should succeed even if (in this example) the TACACS+ server returns an error, enter the following:

```
aaa authentication login default tacacs+ none
```

Note Because the **none** keyword enables *any* user logging in to successfully authenticate, it should be used only as a backup method of authentication.

To create a default list that is used if no list is specified in the **login authentication** command, use the **default** argument followed by the methods you want used in default situations. The default method list is automatically applied to all interfaces.

For example, to specify RADIUS as the default method for user authentication during login, enter the following:

```
aaa authentication login default radius
```

Table 4 lists the supported login authentication methods.

Table 4 AAA Authentication Login Methods

Keyword	Description
enable	Uses the enable password for authentication.
krb5	Uses Kerberos 5 for authentication.
line	Uses the line password for authentication.
local	Uses the local username database for authentication.
none	Uses no authentication.
radius	Uses RADIUS authentication.
tacacs+	Uses TACACS+ authentication.
krb5-telnet	Uses Kerberos 5 Telnet authentication protocol when using Telnet to connect to the router. If selected, this keyword must be listed as the first method in the method list.

Login Authentication Using Local Password

Use the **aaa authentication login** command with the **local method** keyword to specify that the Cisco router or access server will use the local username database for authentication. For example, to specify the local username database as the method of user authentication at login when no other method list has been defined, enter the following:

```
aaa authentication login default local
```

For information about adding users into the local username database, refer to the “Establish Username Authentication” section in this chapter.

Login Authentication Using Line Password

Use the **aaa authentication login** command with the **line method** keyword to specify the line password as the login authentication method. For example, to specify the line password as the method of user authentication at login when no other method list has been defined, enter the following:

```
aaa authentication login default line
```

Before you can use a line password as the login authentication method, you need to define a line password. For more information about defining line passwords, refer to the “Configure Line Password Protection” section in this chapter.

Login Authentication Using Enable Password

Use the **aaa authentication login** command with the **enable method** keyword to specify the enable password as the login authentication method. For example, to specify the enable password as the method of user authentication at login when no other method list has been defined, enter:

```
aaa authentication login default enable
```

Before you can use the enable password as the login authentication method, you need to define the enable password. For more information about defining enable passwords, refer to the “Configuring Passwords and Privileges” chapter.

Login Authentication Using RADIUS

Use the **aaa authentication login** command with the **radius** *method* keyword to specify RADIUS as the login authentication method. For example, to specify RADIUS as the method of user authentication at login when no other method list has been defined, enter:

```
aaa authentication login default radius
```

Before you can use RADIUS as the login authentication method, you need to enable communication with the RADIUS security server. For more information about establishing communication with a RADIUS server, refer to the “Configuring RADIUS” chapter.

Login Authentication Using TACACS+

Use the **aaa authentication login** command with the **tacacs+** *method* keyword to specify TACACS+ as the login authentication method. For example, to specify TACACS+ as the method of user authentication at login when no other method list has been defined, enter:

```
aaa authentication login default tacacs+
```

Before you can use TACACS+ as the login authentication method, you need to enable communication with the TACACS+ security server. For more information about establishing communication with a TACACS+ server, refer to the “Configuring TACACS+” chapter.

Login Authentication Using Kerberos

Authentication via Kerberos is different from most other authentication methods: the user’s password is never sent to the remote access server. Remote users logging in to the network are prompted for a username. If the key distribution center (KDC) has an entry for that user, it creates an encrypted ticket granting ticket (TGT) with the password for that user and sends it back to the router. The user is then prompted for a password, and the router attempts to decrypt the TGT with that password. If it succeeds, the user is authenticated and the TGT is stored in the user’s credential cache on the router.

A user does not need to run the KINIT program to get a TGT to authenticate to the router. This is because KINIT has been integrated into the login procedure in the Cisco IOS implementation of Kerberos.

Use the **aaa authentication login** command with the **krb5** *method* keyword to specify Kerberos as the login authentication method. For example, to specify Kerberos as the method of user authentication at login when no other method list has been defined, enter:

```
aaa authentication login default krb5
```

Before you can use Kerberos as the login authentication method, you need to enable communication with the Kerberos security server. For more information about establishing communication with a Kerberos server, refer to the “Configuring Kerberos” chapter.

Configure PPP Authentication Using AAA

Many users access network access servers through dialup via async or ISDN. Dialup via async or ISDN bypasses the CLI completely; instead, a network protocol (such as PPP or ARAP) starts as soon as the connection is established.

The AAA security services facilitate a variety of authentication methods for use on serial interfaces running PPP. Use the **aaa authentication ppp** command to enable AAA authentication no matter which of the supported PPP authentication methods you decide to use.

To configure AAA authentication methods for serial lines using PPP, perform the following tasks in global configuration mode:

Task	Command
Enable AAA globally.	aaa new-model
Create a local authentication list.	aaa authentication ppp { default <i>list-name</i> } <i>method1</i> [<i>method2...</i>]
Enter interface configuration mode for the interface to which you want to apply the authentication list.	interface <i>interface-type interface-number</i>
Apply the authentication list to a line or set of lines.	ppp authentication { chap pap chap pap pap chap } [if-needed] { default <i>list-name</i> } [callin]

With the **aaa authentication ppp** command, you create one or more lists of authentication methods that are tried when a user tries to authenticate via PPP. These lists are applied using the **ppp authentication** line configuration command.

To create a default list that is used if no list is specified in the **ppp authentication** command, use the **default** argument followed by the methods you want used in default situations.

For example, to specify the local username database as the default method for user authentication, enter the following:

```
aaa authentication ppp default local
```

The keyword *list-name* is any character string used to name the list you are creating. The keyword *method* refers to the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

For example, to specify that authentication should succeed even if (in this example) the TACACS+ server returns an error, enter the following:

```
aaa authentication ppp default tacacs+ none
```

Note Because **none** allows all users logging in to authenticate successfully, it should be used as a backup method of authentication.

Table 5 lists the supported login authentication methods.

Table 5 AAA Authentication PPP Methods

Keyword	Description
if-needed	Does not authenticate if user has already been authenticated on a TTY line.
krb5	Uses Kerberos 5 for authentication (can only be used for PAP authentication).
local	Uses the local username database for authentication.
none	Uses no authentication.
radius	Uses RADIUS authentication.
tacacs+	Uses TACACS+ authentication.

PPP Authentication Using Local Password

Use the **aaa authentication ppp** command with the *method* keyword **local** to specify that the Cisco router or access server will use the local username database for authentication. For example, to specify the local username database as the method of authentication for use on lines running PPP when no other method list has been defined, enter:

```
aaa authentication ppp default local
```

For information about adding users into the local username database, refer to the “Establish Username Authentication” section in this chapter.

PPP Authentication Using RADIUS

Use the **aaa authentication ppp** command with the *radius method* keyword to specify RADIUS as the authentication method for use on interfaces running PPP. For example, to specify RADIUS as the method of user authentication when no other method list has been defined, enter:

```
aaa authentication ppp default radius
```

Before you can use RADIUS as the authentication method, you need to enable communication with the RADIUS security server. For more information about establishing communication with a RADIUS server, refer to the “Configuring RADIUS” chapter.

PPP Authentication Using TACACS+

Use the **aaa authentication ppp** command with the *tacacs+ method* keyword to specify TACACS+ as the authentication method for use on interfaces running PPP. For example, to specify TACACS+ as the method of user authentication when no other method list has been defined, enter:

```
aaa authentication ppp default tacacs+
```

Before you can use TACACS+ as the authentication method, you need to enable communication with the TACACS+ security server. For more information about establishing communication with a TACACS+ server, refer to the “Configuring TACACS+” chapter.

PPP Authentication Using Kerberos

Use the **aaa authentication ppp** command with the **krb5** *method* keyword to specify Kerberos as the authentication method for use on interfaces running PPP. For example, to specify Kerberos as the method of user authentication when no other method list has been defined, enter:

```
aaa authentication ppp default krb5
```

Before you can use Kerberos as the login authentication method, you need to enable communication with the Kerberos security server. For more information about establishing communication with a Kerberos server, refer to the “Configuring Kerberos” chapter.

Note Kerberos login authentication works only with PPP PAP authentication.

Configure ARA Authentication Using AAA

With the **aaa authentication arap** command, you create one or more lists of authentication methods that are tried when AppleTalk Remote Access (ARA) users attempt to log in to the router. These lists are used with the **arap authentication** line configuration command.

Perform at least the first of the following tasks starting in global configuration mode:

Task	Command
Enable AAA globally.	aaa new-model
Enable authentication for ARA users.	aaa authentication arap { default <i>list-name</i> } <i>method1</i> [<i>method2...</i>]
(Optional) Change to line configuration mode.	line <i>number</i>
(Optional) Enable autoselection of ARA.	autoselect arap
(Optional) Start the ARA session automatically at user login.	autoselect during-login
(Optional—not needed if default is used in the aaa authentication arap command) Enable TACACS+ authentication for ARA on a line.	arap authentication <i>list-name</i>

The *list-name* is any character string used to name the list you are creating. The *method* refers to the actual list of methods the authentication algorithm tries, in the sequence entered.

To create a default list that is used if no list is specified in the **arap authentication** command, use the **default** argument followed by the methods you want to be used in default situations.

The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

Note Because **none** allows all users logging in to authenticate successfully, it should be used as a backup method of authentication.

Table 6 lists the supported login authentication methods.

Table 6 AAA Authentication ARAP Methods

Keyword	Description
guest	Allows guest logins.
auth-guest	Allows guest logins only if the user has already logged into EXEC.
line	Uses the line password for authentication.
local	Uses the local username database for authentication.
tacacs+	Uses TACACS+ authentication.

For example, to create a default AAA authentication method list used with the ARA protocol, enter:

```
aaa authentication arap default if-needed none
```

To create the same authentication method list for the ARA protocol but name the list *MIS-access*, enter:

```
aaa authentication arap MIS-access if-needed none
```

ARA Authentication Allowing Guest Logins

Use the **aaa authentication arap** command with the **guest** keyword to allow guest logins. This method must be the first listed in the ARA authentication method list but it can be followed by other methods if it does not succeed. For example, to allow all guest logins as the default method of authentication, using TACACS+ only if that method fails, enter:

```
aaa authentication arap default guest tacacs+
```

For more information about ARA guest logins, refer to the “Configuring AppleTalk” chapter in the *Network Protocols Configuration Guide, Part 2*.

ARA Authentication Allowing Authorized Guest Logins

Use the **aaa authentication arap** command with the **auth-guest** keyword to allow guest logins only if the user has already successfully logged in to the EXEC. This method must first in the ARA authentication method list but it can be followed by other methods if it does not succeed. For example, to allow all authorized guest logins—meaning logins by users who have already successfully logged into the EXEC—as the default method of authentication, using TACACS+ only if that method fails, enter:

```
aaa authentication arap default auth-guest tacacs+
```

For more information about ARA authorized guest logins, refer to the “Configuring AppleTalk” chapter in the *Network Protocols Configuration Guide, Part 2*.

Note By default, guest logins through ARAP are disabled when you initialize AAA. To allow guest logins, you must use the **aaa authentication arap** command with either the **guest** or **auth-guest** keyword.

ARA Authentication Using Local Password

Use the **aaa authentication arap** command with the *method* keyword **local** to specify that the Cisco router or access server will use the local username database for authentication. For example, to specify the local username database as the method of ARA user authentication when no other method list has been defined, enter:

```
aaa authentication arap default local
```

For information about adding users to the local username database, refer to the “Establish Username Authentication” section in this chapter.

ARA Authentication Using Line Password

Use the **aaa authentication arap** command with the *method* keyword **line** to specify the line password as the authentication method. For example, to specify the line password as the method of ARA user authentication when no other method list has been defined, enter:

```
aaa authentication arap default line
```

Before you can use a line password as the ARA authentication method, you need to define a line password. For more information about defining line passwords, refer to the “Configure Line Password Protection” section in this chapter.

ARA Authentication Using TACACS+

Use the **aaa authentication arap** command with the **tacacs+** *method* keyword to specify TACACS+ as the ARA authentication method. For example, to specify TACACS+ as the method of ARA user authentication when no other method list has been defined, enter:

```
aaa authentication arap default tacacs+
```

Before you can use TACACS+ as the ARA authentication method, you need to enable communication with the TACACS+ security server. For more information about establishing communication with a TACACS+ server, refer to the “Configuring TACACS+” chapter.

Configure NASI Authentication Using AAA

With the **aaa authentication nasi** command, you create one or more lists of authentication methods that are tried when Netware Asynchronous Services Interface (NASI) users attempt to log in to the router. These lists are used with the **nasi authentication** line configuration command.

Perform at least the first of the following tasks starting in global configuration mode:

Task	Command
Enable AAA globally.	aaa new-model
Enable authentication for NASI users.	aaa authentication nasi { default <i>list-name</i> } <i>method1</i> [<i>method2...</i>]
(Optional—not needed if default is used in the aaa authentication nasi command.) Change to line configuration mode.	line number
(Optional—not needed if default is used in the aaa authentication nasi command) Enable authentication for NASI on a line.	nasi authentication list-name

The *list-name* is any character string used to name the list you are creating. The *method* refers to the actual list of methods the authentication algorithm tries, in the sequence entered.

To create a default list that is used if no list is specified in the **aaa authentication nasi** command, use the **default** argument followed by the methods you want to be used in default situations.

The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

Note Because **none** allows all users logging in to authenticate successfully, it should be used as a backup method of authentication.

Table 7 lists the supported login authentication methods.

Table 7 AAA Authentication NASI Methods

Keyword	Description
enable	Uses the enable password for authentication.
line	Uses the line password for authentication.
local	Uses the local username database for authentication.
none	Uses no authentication.
tacacs+	Uses TACACS+ authentication.

NASI Authentication Using Enable Password

Use the **aaa authentication nasi** command with the argument **enable** to specify the enable password as the authentication method. For example, to specify the enable password as the method of NASI user authentication when no other method list has been defined, enter:

```
aaa authentication nasi default enable
```

Before you can use the enable password as the authentication method, you need to define the enable password. For more information about defining enable passwords, refer to the “Configuring Passwords and Privileges” chapter.

NASI Authentication Using Local Password

Use the **aaa authentication nasi** command with the *method* keyword **local** to specify that the Cisco router or access server will use the local username database for authentication information. For example, to specify the local username database as the method of NASI user authentication when no other method list has been defined, enter:

```
aaa authentication nasi default local
```

For information about adding users to the local username database, refer to the “Establish Username Authentication” section in this chapter.

NASI Authentication Using Line Password

Use the **aaa authentication nasi** command with the *method* keyword **line** to specify the line password as the authentication method. For example, to specify the line password as the method of NASI user authentication when no other method list has been defined, enter:

```
aaa authentication nasi default line
```

Before you can use a line password as the NASI authentication method, you need to define a line password. For more information about defining line passwords, refer to the “Configure Line Password Protection” section in this chapter.

NASI Authentication Using TACACS+

Use the **aaa authentication nasi** command with the **tacacs+** *method* keyword to specify TACACS+ as the NASI authentication method. For example, to specify TACACS+ as the method of NASI user authentication when no other method list has been defined, enter:

```
aaa authentication nasi default tacacs+
```

Before you can use TACACS+ as the authentication method, you need to enable communication with the TACACS+ security server. For more information about establishing communication with a TACACS+ server, refer to the “Configuring TACACS+” chapter.

Enable Password Protection at the Privileged Level

Use the **aaa authentication enable default** command to create a series of authentication methods that are used to determine whether a user can access the privileged EXEC command level. You can specify up to four authentication methods. The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication succeed even if all methods return an error, specify **none** as the final method in the command line.

Perform the following task in global configuration mode:

Task	Command
Enable user ID and password checking for users requesting privileged EXEC level.	aaa authentication enable default <i>method1</i> [<i>method2...</i>]

The *method* refers to the actual list of methods the authentication algorithm tries, in the sequence entered. Table 8 lists the supported login authentication methods.

Table 8 AAA Authentication Enable Default Methods

Keyword	Description
enable	Uses the enable password for authentication.
line	Uses the line password for authentication.
none	Uses no authentication.
tacacs+	Uses TACACS+ authentication.
radius	Uses RADIUS authentication.

Enable an Authentication Override

To configure the Cisco IOS software to check the local user database for authentication before attempting another form of authentication, use the **aaa authentication local-override** command. This command is useful when you want to configure an override to the normal authentication process for certain personnel (such as system administrators).

Perform the following task in global configuration mode:

Task	Command
Create an override for authentication.	aaa authentication local-override

Change the Text Displayed at the Password Prompt

Use the **aaa authentication password-prompt** command to change the default text that the Cisco IOS software displays when prompting a user to enter a password. This command changes the password prompt for the enable password as well as for login passwords that are not supplied by remote security servers. The **no** form of this command returns the password prompt to the following default value:

```
Password:
```

The **aaa authentication password-prompt** command does not change any dialog that is supplied by a remote TACACS+ server.

The **aaa authentication password-prompt** command works when using RADIUS as the login method. You will be able to see the password prompt defined in the command shown even when the RADIUS server is unreachable. The **aaa authentication password-prompt** command does not work with TACACS+. TACACS+ supplies the NAS the password prompt to display to the users. If the TACACS+ server is reachable, the NAS gets the password prompt from the server and uses that prompt instead of the one defined in the **aaa authentication password-prompt** command. If the TACACS+ server is not reachable, the password prompt defined in the **aaa authentication password-prompt** command may be used.

Perform the following task in global configuration mode:

Task	Command
Change the default text displayed when a user is prompted to enter a password.	aaa authentication password-prompt <i>text-string</i>

Enable Double Authentication

Double Authentication provides additional authentication for Point-to-Point Protocol (PPP) sessions. Previously, PPP sessions could only be authenticated by using a single authentication method: either PAP or CHAP. Double Authentication requires remote users to pass a second stage of authentication—after CHAP or PAP authentication—before gaining network access.

This second (“double”) authentication requires a password that is known to the user but *not* stored on the user’s remote host. Therefore, the second authentication is specific to a user, not to a host. This provides an additional level of security that will be effective even if information from the remote host is stolen. In addition, this also provides greater flexibility by allowing customized network privileges for each user.

The second stage authentication can use one-time passwords such as token card passwords, which are not supported by CHAP. If one-time passwords are used, a stolen user password is of no use to the perpetrator.

How Double Authentication Works

With Double Authentication, there are two authentication/authorization stages. These two stages occur after a remote user dials in and a PPP session is initiated.

In the first stage, the user logs in using the remote host name; CHAP (or PAP) authenticates the remote host, and then PPP negotiates with AAA to authorize the remote host. In this process, the network access privileges associated with the remote host are assigned to the user.

Note We suggest that the network administrator restrict authorization at this first stage to allow only Telnet connections to the local host.

In the second stage, the remote user must Telnet to the network access server to be authenticated. When the remote user logs in, the user must be authenticated with AAA login authentication. The user then must enter the **access-profile** command to be reauthorized using AAA. When this authorization is complete, the user has been double authenticated, and can access the network according to per-user network privileges.

The system administrator determines what network privileges remote users will have after each stage of authentication by configuring appropriate parameters on a security server. To use Double Authentication, the user must activate it by issuing the **access-profile** command.

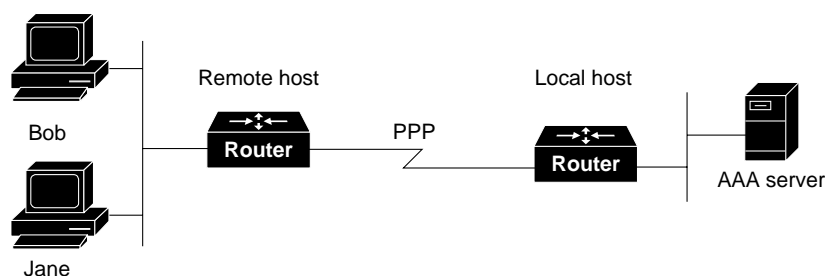


Caution Double Authentication can cause certain undesirable events if multiple hosts share a PPP connection to a network access server, as shown in Figure 4.

First, if a user, Bob, initiates a PPP session and activates Double Authentication at the NAS (per Figure 4), any other user will automatically have the same network privileges as Bob until Bob's PPP session expires. This happens because Bob's authorization profile is applied to the NAS's interface during the PPP session and any PPP traffic from other users will use the PPP session Bob established.

Second, if Bob initiates a PPP session and activates Double Authentication, and then—before Bob's PPP session has expired—another user, Jane, executes the **access-profile** command (or, if she Telnets to the NAS and **autocommand access-profile** is executed), a reauthorization will occur and Jane's authorization profile will be applied to the interface—replacing Bob's profile. This can disrupt or halt Bob's PPP traffic, or grant Bob additional authorization privileges he should not have.

Figure 4 Possibly Risky Topology: Multiple Hosts Share a PPP Connection to a NAS



55923

Configure Double Authentication

To configure Double Authentication, you must complete the following steps:

- 1 Enable AAA by using the **aaa-new model** global configuration command. For more information about enabling AAA, refer to the “AAA Overview” chapter.
- 2 Use the **aaa authentication** command to configure your network access server to use login and PPP authentication method lists, then apply those method lists to the appropriate lines or interfaces.
- 3 Use the **aaa authorization** command to configure AAA network authorization at login. For more information about configuring network authorization, refer to the “Configuring Authorization” chapter.
- 4 Configure security protocol parameters (for example, RADIUS or TACACS+). For more information about RADIUS, refer to the “Configuring RADIUS” chapter. For more information about TACACS+, refer to the “Configuring TACACS+” chapter.
- 5 Use access control list AV pairs on the security server that the user can connect to the local host only by establishing a Telnet connection.
- 6 (Optional) Configure the **access-profile** command as an autocommand. If you configure the autocommand, remote users will not have to manually enter the **access-profile** command to access authorized rights associated with their personal user profile. To learn about configuring autocommands, refer to the **autocommand** command in the *Dial Solutions Command Reference*.

Note If the **access-profile** command is configured as an autocommand, users will still have to Telnet to the local host and log in to complete Double Authentication.

Follow these rules when creating the user-specific authorization statements (These rules relate to the default behavior of the **access-profile** command):

- Use valid AV pairs when configuring access control list AV pairs on the security server. For a list of valid AV pairs, refer to the “Authentication Commands” chapter in the *Security Command Reference*.
- If you want remote users to use the interface’s existing authorization (that which existed prior to the second stage authentication/authorization), but you want them to have different access control lists (ACLs), you should specify *only* ACL AV pairs in the user-specific authorization definition. This might be desirable if you set up a default authorization profile to apply to the remote host, but want to apply specific ACLs to specific users.
- When these user-specific authorization statements are later applied to the interface, they can either be *added* to the existing interface configuration, or *replace* the existing interface configuration—depending on which form of the **access-profile** command is used to authorize the user. You should understand how the **access-profile** command works before configuring the authorization statements.
- If you will be using ISDN or Multilink PPP, you must also configure virtual templates at the local host.

To troubleshoot Double Authentication, use the **debug aaa per-user** debug command. For more information about this command, refer to the *Debug Command Reference*.

Access User Profile after Double Authentication

In Double Authentication, when a remote user establishes a PPP link to the local host using the local host name, the remote host is CHAP (or PAP) authenticated. After CHAP (or PAP) authentication, PPP negotiates with AAA to assign network access privileges associated with the remote host to the user. (We suggest that privileges at this stage be restricted to allow the user to connect to the local host only by establishing a Telnet connection.)

When the user needs to initiate the second phase of Double Authentication, establishing a Telnet connection to the local host, the user enters a personal username and password (different from the CHAP or PAP username and password). This action causes AAA reauthentication to occur according to the personal username/password. The initial rights associated with the local host, though, are still in place. By using the **access-profile** command, the rights associated with the local host are replaced by or merged with those defined for the user in the user's profile.

To access the rights associated with the user after Double Authentication, perform the following tasks in EXEC configuration mode:

Task	Command
Access the rights associated for the user after Double Authentication.	access profile [merge replace ignore-sanity-checks]

If you configured the **access-profile** command to be executed as an autocommand, it will be executed automatically after the remote user logs in.

Non-AAA Authentication Methods

This section discusses the following non-AAA authentication tasks:

- Configure Line Password Protection
- Establish Username Authentication
- Enable CHAP or PAP Authentication
- Configure TACACS and Extended TACACS Password Protection

Configure Line Password Protection

You can provide access control on a terminal line by entering the password and establishing password checking. To do so, perform the following tasks in line configuration mode:

Task	Command
Assign a password to a terminal or other device on a line.	password <i>password</i>
Enable password checking at login.	login

The password checker is case sensitive and can include spaces; for example, the password “Secret” is different from the password “secret,” and “two words” is an acceptable password.

You can disable line password verification by disabling password checking. To do so, perform the following task in line configuration mode:

Task	Command
Disable password checking or allow access to a line without password verification.	no login

If you configure line password protection and then configure TACACS or Extended TACACS, the TACACS username and password take precedence over line passwords. If you have not yet implemented a security policy, we recommend that you use AAA.

Establish Username Authentication

You can create a username-based authentication system, which is useful in the following situations:

- To provide a TACACS-like username and encrypted password-authentication system for networks that cannot support TACACS
- To provide special-case logins; for example, access list verification, no password verification, autocommand execution at login, and “no escape” situation

To establish username authentication, perform the following tasks in global configuration mode as needed for your system configuration:

Task	Command
Establish username authentication with encrypted passwords. or (Optional) Establish username authentication by access list.	username <i>name</i> [no password password <i>encryption-type</i> password]
(Optional) Set the privilege level for the user.	username <i>name</i> privilege <i>level</i>
(Optional) Specify a command to automatically execute.	username <i>name</i> [autocommand <i>command</i>]
(Optional) Set a “no escape” login environment.	username <i>name</i> [noescape] [nohangup]

The keyword **noescape** prevents users from using escape characters on the hosts to which they are connected. The **nohangup** feature does not disconnect after using the autocommand.



Caution Passwords will be displayed in clear text in your configuration unless you enable the **service password-encryption** command. For more information about the **service password-encryption** command, refer to the “Password and Privileges Commands” chapter in the *Security Command Reference*.

Enable CHAP or PAP Authentication

One of the most common transport protocols used in Internet Service Providers’ (ISPs’) dial solutions is the Point-to-Point Protocol (PPP). Traditionally, remote users dial in to an access server to initiate a PPP session. After PPP has been negotiated, remote users are connected to the ISP network and to the Internet.

Because ISPs want only customers to connect to their access servers, remote users are required to authenticate to the access server before they can start up a PPP session. Normally, a remote user authenticates by typing in a username and password when prompted by the access server. Although this is a workable solution, it is difficult to administer and awkward for the remote user.

A better solution is to use the authentication protocols built into PPP. In this case, the remote user dials in to the access server and starts up a minimal subset of PPP with the access server. This does not give the remote user access to the ISP’s network—it merely allows the access server to talk to the remote device.

PPP currently supports two authentication protocols: Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP). Both are specified in RFC 1334 and are supported on synchronous and asynchronous interfaces. Authentication via PAP or CHAP is equivalent to typing in a username and password when prompted by the server. CHAP is considered to be more secure because the remote user’s password is never sent across the connection.

PPP (with or without PAP or CHAP authentication) is also supported in dialout solutions. An access server utilizes a dialout feature when it initiates a call to a remote device and attempts to start up a transport protocol such as PPP.

See the “Configuring Interfaces” chapter in the *Configuration Fundamentals Configuration Guide* for more information about CHAP and PAP.

Note To use CHAP or PAP, you must be running PPP encapsulation.

When CHAP is enabled on an interface and a remote device attempts to connect to it, the access server sends a CHAP packet to the remote device. The CHAP packet requests or “challenges” the remote device to respond. The challenge packet consists of an ID, a random number, and the host name of the local router.

When the remote device receives the challenge packet, it concatenates the ID, the remote device’s password, and the random number, and then encrypts all of it using the remote device’s password. The remote device sends the results back to the access server, along with the name associated with the password used in the encryption process.

When the access server receives the response, it uses the name it received to retrieve a password stored in its user database. The retrieved password should be the same password the remote device used in its encryption process. The access server then encrypts the concatenated information with the newly-retrieved password—if the result matches the result sent in the response packet, authentication succeeds.

The benefit of using CHAP authentication is that the remote device’s password is never transmitted in clear text. This prevents other devices from stealing it and gaining illegal access to the ISP’s network.

CHAP transactions occur only at the time a link is established. The access server does not request a password during the rest of the call. (The local device can, however, respond to such requests from other devices during a call.)

When PAP is enabled, the remote router attempting to connect to the access server is required to send an authentication request. If the username and password specified in the authentication request are accepted, the Cisco IOS software sends an authentication acknowledgment.

After you have enabled CHAP or PAP, the access server will require authentication from remote devices dialing in to the access server. If the remote device does not support the enabled protocol, the call will be dropped.

To use CHAP or PAP, you must perform the following tasks:

- 1 Enable PPP encapsulation.
- 2 Enable CHAP or PAP on the interface.
- 3 For CHAP, configure host name authentication and the secret or password for each remote system with which authentication is required.

Enable PPP Encapsulation

To enable PPP encapsulation, perform the following task in interface configuration mode:

Task	Command
Enable PPP on an interface.	encapsulation ppp

Enable PAP or CHAP

To enable CHAP or PAP authentication on an interface configured for PPP encapsulation, perform the following task in interface configuration mode:

Task	Command
Define the authentication methods supported and the order in which they are used.	ppp authentication { chap chap pap pap chap pap } [if-needed] [<i>list-name</i> default] [callin] [one-time]

If you configure **ppp authentication chap** on an interface, all incoming calls on that interface that initiate a PPP connection will have to be authenticated using CHAP; likewise, if you configure **ppp authentication pap**, all incoming calls that start a PPP connection will have to be authenticated via PAP. If you configure **ppp authentication chap pap**, the access server will attempt to authenticate all incoming calls that start a PPP session with CHAP. If the remote device does not support CHAP, the access server will try to authenticate the call using PAP. If the remote device doesn't support either CHAP or PAP, authentication will fail and the call will be dropped. If you configure **ppp authentication pap chap**, the access server will attempt to authenticate all incoming calls that start a PPP session with PAP. If the remote device does not support PAP, the access server will try to authenticate the call using CHAP. If the remote device doesn't support either protocols, authentication will fail and the call will be dropped. If you configure the **ppp authentication** command with the **callin** keyword, the access server will only authenticate the remote device if the remote device initiated the call.

Authentication method lists and the **one-time** keyword are only available if you have enabled AAA—they will not be available if you are using TACACS or Extended TACACS. If you specify the name of an authentication method list with the **ppp authentication** command, PPP will attempt to authenticate the connection using the methods defined in the specified method list. If AAA is enabled and no method list is defined by name, PPP will attempt to authenticate the connection using the methods defined as the default. The **ppp authentication** command with the **one-time** keyword enables support for one-time passwords during authentication.

The **if-needed** keyword is only available if you are using TACACS or Extended TACACS. The **ppp authentication** command with the **if-needed** keyword means that PPP will only authenticate the remote device via PAP or CHAP if they have not yet authenticated during the life of the current call. If the remote device authenticated via a standard login procedure and initiated PPP from the EXEC prompt, PPP will not authenticate via CHAP if **ppp authentication chap if-needed** is configured on the interface.



Caution If you use a *list-name* that has not been configured with the **aaa authentication ppp** command, you disable PPP on the line.

For information about adding a **username** entry for each remote system from which the local router or access server requires authentication, see the “Establish Username Authentication” section.

Inbound and Outbound Authentication

PPP supports two-way authentication. Normally, when a remote device dials in to an access server, the access server requests that the remote device prove that it is allowed access. This is known as inbound authentication. At the same time, the remote device can also request that the access server prove that it is who it says it is. This is known as outbound authentication. An access server also does outbound authentication when it initiates a call to a remote device.

Enabling Outbound PAP Authentication

To enable outbound PAP authentication, perform the following task in interface configuration mode:

Task	Command
Enable outbound PAP authentication.	ppp pap sent-username <i>username</i> password <i>password</i>

The access server uses the username and password specified by the **ppp pap sent-username** command to authenticate itself whenever it initiates a call to a remote device or when it has to respond to a remote device's request for outbound authentication.

Create a Common CHAP Password

For remote CHAP authentication only, you can configure your router to create a common CHAP secret password to use in response to challenges from an unknown peer; for example, if your router calls a rotary of routers (either from another vendor, or running an older version of the Cisco IOS software) to which a new (that is, unknown) router has been added. The **ppp chap password** command allows you to replace several username and password configuration commands with a single copy of this command on any dialer interface or asynchronous group interface.

To enable a router calling a collection of routers to configure a common CHAP secret password, perform the following task in interface configuration mode:

Task	Command
Enable a router calling a collection of routers to configure a common CHAP secret password.	ppp chap password <i>secret</i>

Refuse CHAP Authentication Requests

To refuse CHAP authentication from peers requesting it, meaning that CHAP authentication is disabled for all calls, perform the following task in interface configuration mode:

Task	Command
Refuse CHAP authentication from peers requesting CHAP authentication.	ppp chap refuse [callin]

If the **callin** keyword is used, the router will refuse to answer CHAP authentication challenges received from the peer, but will still require the peer to answer any CHAP challenges the router sends.

If outbound PAP has been enabled (using the **ppp pap sent-username** command), PAP will be suggested as the authentication method in the refusal packet.

Delay CHAP Authentication Until Peer Authenticates

To specify that the router will not authenticate to a peer requesting CHAP authentication until after the peer has authenticated itself to the router, perform the following task in interface configuration mode:

Task	Command
Configure the router to delay CHAP authentication until after the peer has authenticated itself to the router.	ppp chap wait <i>secret</i>

This command (which is the default) specifies that the router will not authenticate to a peer requesting CHAP authentication until the peer has authenticated itself to the router. The **no ppp chap wait** command specifies that the router will respond immediately to an authentication challenge.

Configure TACACS and Extended TACACS Password Protection

You can use TACACS or Extended TACACS to control login access to the router. Perform the tasks in the following sections:

- Set TACACS Password Protection at the User Level
- Disable Password Checking at the User Level

Before performing these tasks, you must have enabled communication with a TACACS host on the network. For more information, refer to the “Configuring TACACS and Extended TACACS” chapter in the *Security Configuration Guide*.

Set TACACS Password Protection at the User Level

You can enable TACACS password checking at login by performing the following task in line configuration mode:

Task	Command
Set the TACACS-style user ID and password-checking mechanism.	login tacacs

Disable Password Checking at the User Level

If a TACACS server does not respond to a login request, the Cisco IOS software denies the request by default. However, you can prevent login failure in one of two ways:

- Allow a user to access privileged EXEC mode if that user enters the password set by the **enable** command.
- Ensure a successful login by allowing the user to access the privileged EXEC mode without further question.

To specify one of these features, perform either of the following tasks in global configuration mode:

Task	Command
Allow a user to access privileged EXEC mode, or set last resort options for logins.	tacacs-server last-resort password or tacacs-server last-resort succeed

Authentication Examples

This section contains the following authentication configuration examples:

- RADIUS Authentication Examples
- TACACS+ Authentication Examples
- TACACS and Extended TACACS Authentication Examples
- Kerberos Authentication Examples
- Double Authentication Configuration Example

RADIUS Authentication Examples

This section provides two sample configurations using RADIUS.

The following example shows how to configure the router to authenticate and authorize using RADIUS:

```
aaa authentication login radius-login RADIUS local
aaa authentication ppp radius-ppp if-needed radius
aaa authorization exec radius if-authenticated
aaa authorization network radius
line 3
login authentication radius-login
interface serial 0
ppp authentication radius-ppp
```

The lines in this sample RADIUS authentication and authorization configuration are defined as follows:

- The **aaa authentication login radius-login RADIUS local** command configures the router to use RADIUS for authentication at the login prompt. If RADIUS returns an error, the user is authenticated using the local database.
- The **aaa authentication ppp radius-ppp if-needed radius** command configures the Cisco IOS software to use PPP authentication using CHAP or PAP if the user has not already logged in. If the EXEC facility has authenticated the user, PPP authentication is not performed.
- The **aaa authorization exec radius if-authenticated** command queries the RADIUS database for information that is used during EXEC authorization, such as autocommands and privilege levels, but only provides authorization if the user has successfully authenticated.
- The **aaa authorization network radius** command queries RADIUS for network authorization, address assignment, and other access lists.
- The **login authentication radius-login** command enables the use-radius method list for line 3.
- The **ppp authentication radius-ppp** command enables the user-radius method list for serial interface 0.

The following example shows how to configure the router to prompt for and verify a username and password, authorize the user's EXEC level, and specify it as the method of authorization for privilege level 2. In this example, if a local username is entered at the username prompt, that username is used for authentication.

If the user is authenticated using the local database, EXEC authorization using RADIUS will fail because no data is saved from the RADIUS authentication. The method list also uses the local database to find an autocommand. If there is no autocommand, the user becomes the EXEC user. If the user then attempts to issue commands that are set at privilege level 2, TACACS+ is used to attempt to authorize the command.

```
aaa authentication local-override
aaa authentication login default radius local
aaa authorization exec radius local
aaa authorization command 2 tacacs+ if-authenticated
```

The lines in this sample RADIUS authentication and authorization configuration are defined as follows:

- The **aaa authentication local-override** command specifies that the username prompt appear before authentication starts, and that the authentication always use the local database if the user has a local account.
- The **aaa authentication login default radius local** command specifies that the username and password are verified by RADIUS or, if RADIUS is not responding, by the router's local user database.
- The **aaa authorization exec radius local** command specifies that RADIUS authentication information be used to set the user's EXEC level if the user authenticates with RADIUS. If no RADIUS information is used, this command specifies that the local user database be used for EXEC authorization.
- The **aaa authorization command 2 tacacs+ if-authenticated** command specifies TACACS+ authorization for commands set at privilege level 2, if the user has already successfully authenticated.

TACACS+ Authentication Examples

The following example configures TACACS+ as the security protocol to be used for PPP authentication:

```
aaa new-model
aaa authentication ppp test tacacs+ local
interface serial 0
ppp authentication chap pap test
tacacs-server host 10.1.2.3
tacacs-server key goaway
```

The lines in this sample TACACS+ authentication configuration are defined as follows:

- The **aaa new-model** command enables the AAA security services.
- The **aaa authentication** command defines a method list, "test," to be used on serial interfaces running PPP. The keyword **tacacs+** means that authentication will be done through TACACS+. If TACACS+ returns an ERROR of some sort during authentication, the keyword **local** indicates that authentication will be attempted using the local database on the network access server.
- The **interface** command selects the line.
- The **ppp authentication** command applies the test method list to this line.
- The **tacacs-server host** command identifies the TACACS+ daemon as having an IP address of 10.1.2.3.
- The **tacacs-server key** command defines the shared encryption key to be "goaway."

The following example configures AAA authentication for PPP:

```
aaa authentication ppp default if-needed tacacs+ local
```

In this example, the keyword **default** means that PPP authentication is applied by default to all interfaces. The **if-needed** keyword means that if the user has already authenticated by going through the ASCII login procedure, then PPP is not necessary and can be skipped. If authentication is needed, the keyword **tacacs+** means that authentication will be done through TACACS+. If TACACS+ returns an ERROR of some sort during authentication, the keyword **local** indicates that authentication will be attempted using the local database on the network access server.

The following example creates the same authentication algorithm for PAP but calls the method list “MIS-access” instead of “default”:

```
aaa authentication pap MIS-access if-needed tacacs+ local
interface serial 0
ppp authentication MIS-access
```

In this example, since the list does not apply to any interfaces (unlike the default list, which applies automatically to all interfaces), the administrator must select interfaces to which this authentication scheme should apply by using the **interface** command. The administrator must then apply this method list to those interfaces by using the **ppp authentication** command.

TACACS and Extended TACACS Authentication Examples

The following example shows TACACS enabled for PPP authentication:

```
int async 1
ppp authentication chap
ppp use-tacacs
```

The following example shows TACACS enabled for ARAP authentication:

```
line 3
arap use-tacacs
```

Kerberos Authentication Examples

To specify Kerberos as the authentication method, use the following command:

```
aaa authentication login default krb5
```

Use the following command to specify Kerberos authentication for PPP:

```
aaa authentication ppp default krb5
```

Double Authentication Configuration Example

The examples in this section illustrate possible configurations to be used with Double Authentication. Your configurations could differ significantly, depending on your network and security requirements.

These examples are included:

- Configuring the Local Host for AAA with Double Authentication Examples
- Configuring the AAA Server for First-Stage (PPP) Authentication/Authorization Example
- Configuring the AAA Server for Second-Stage (Per-User) Authentication/Authorization Examples
- Complete Sample Configuration with TACACS+

Note These configuration examples include specific IP addresses and other specific information. This information is for illustration purposes only: your configuration will use different IP addresses, different usernames and passwords, and different authorization statements.

Configuring the Local Host for AAA with Double Authentication Examples

These two examples configure a local host to use AAA for PPP and login authentication, and for network and EXEC authorization. One example is shown for RADIUS and one example for TACACS+.

In both examples, the first three lines configure AAA, with a specific server as the AAA server. The next two lines configure AAA for PPP and login authentication, and the last two lines configure network and EXEC authorization. The last line is necessary only if the **access-profile** command will be executed as an autocommand.

Example Router Configuration with a RADIUS AAA

```
aaa new-model
radius-server host secureserver
radius-server key myradiuskey
aaa authentication ppp default radius
aaa authentication login default radius
aaa authorization network radius
aaa authorization exec radius
```

Example Router Configuration with a TACACS+ Server

```
aaa new-model
tacacs-server host security
tacacs-server key mytacacskey
aaa authentication ppp default tacacs+
aaa authentication login default tacacs+
aaa authorization network tacacs+
aaa authorization exec tacacs+
```

Configuring the AAA Server for First-Stage (PPP) Authentication/Authorization Example

This example shows a configuration on the AAA server. A partial sample AAA configuration is shown for RADIUS.

TACACS+ servers can be configured similarly. (See the “Complete Sample Configuration with TACACS+” section later in this document.)

This example defines authentication/authorization for a remote host named “hostx” that will be authenticated by CHAP in the first stage of Double Authentication. Note that the ACL AV pair limits the remote host to Telnet connections to the local host. The local host has the IP address 10.0.0.2.

Example RADIUS AAA Server Configuration

```
hostx Password = "welcome"
      User-Service-Type = Framed-User,
      Framed-Protocol = PPP,
      cisco-avpair = "lcp:interface-config=ip unnumbered ethernet 0",
      cisco-avpair = "ip:inacl#3=permit tcp any 172.21.114.0 0.0.0.255 eq telnet",
      cisco-avpair = "ip:inacl#4=deny icmp any any",
      cisco-avpair = "ip:route#5=55.0.0.0 255.0.0.0",
      cisco-avpair = "ip:route#6=66.0.0.0 255.0.0.0",
      cisco-avpair = "ipx:inacl#3=deny any"
```

Configuring the AAA Server for Second-Stage (Per-User) Authentication/Authorization Examples

This section contains partial sample AAA configurations on a RADIUS server. These configurations define authentication/authorization for a user (Bob) with the username "bobuser," who will be user-authenticated in the second stage of Double Authentication.

TACACS+ servers can be configured similarly. (See the "Complete Sample Configuration with TACACS+" section later in this document.)

Example RADIUS AAA Server Configurations

Three examples show sample RADIUS AAA configurations that could be used with each of the three forms of the **access-profile** command.

The first example shows a partial sample AAA configuration that works with the default form (no keywords) of the **access-profile** command. Note that only ACL AV pairs are defined. This example also sets up the **access-profile** command as an autocommand.

```
bobuser Password = "welcome"
        User-Service-Type = Shell-User,
        cisco-avpair = "shell:autocmd=access-profile"
        User-Service-Type = Framed-User,
        Framed-Protocol = PPP,
        cisco-avpair = "ip:inacl#3=permit tcp any host 10.0.0.2 eq telnet",
        cisco-avpair = "ip:inacl#4=deny icmp any any"
```

The second example shows a partial sample AAA configuration that works with the **access-profile merge** form of the **access-profile** command. This example also sets up the **access-profile merge** command as an autocommand.

```
bobuser Password = "welcome"
        User-Service-Type = Shell-User,
        cisco-avpair = "shell:autocmd=access-profile merge"
        User-Service-Type = Framed-User,
        Framed-Protocol = PPP,
        cisco-avpair = "ip:inacl#3=permit tcp any any"
        cisco-avpair = "ip:route=10.0.0.0 255.255.0.0",
        cisco-avpair = "ip:route=10.1.0.0 255.255.0.0",
        cisco-avpair = "ip:route=10.2.0.0 255.255.0.0"
```

The third example shows a partial sample AAA configuration that works with the **access-profile replace** form of the **access-profile** command. This example also sets up the **access-profile replace** command as an autocommand.

```

bobuser          Password = "welcome"
                 User-Service-Type = Shell-User,
                 cisco-avpair = "shell:autocmd=access-profile replace"
                 User-Service-Type = Framed-User,
                 Framed-Protocol = PPP,
                 cisco-avpair = "ip:inacl#3=permit tcp any any",
                 cisco-avpair = "ip:inacl#4=permit icmp any any",
                 cisco-avpair = "ip:route=10.10.0.0 255.255.0.0",
                 cisco-avpair = "ip:route=10.11.0.0 255.255.0.0",
                 cisco-avpair = "ip:route=10.12.0.0 255.255.0.0"

```

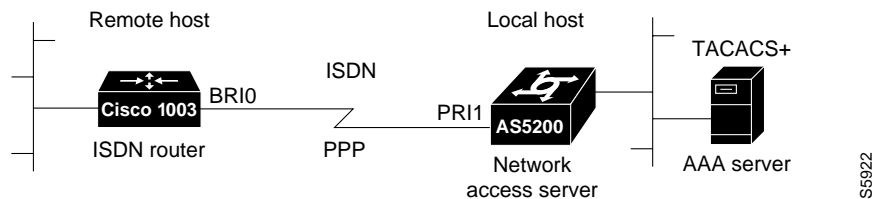
Complete Sample Configuration with TACACS+

This example shows TACACS+ authorization profile configurations both for the remote host (used in the first stage of Double Authentication) and for specific users (used in the second stage of Double Authentication). This TACACS+ example contains approximately the same configuration information as shown in the previous RADIUS examples.

This sample configuration shows authentication/authorization profiles on the TACACS+ server for the remote host "hostx" and for three users, with the usernames "bob_default," "bob_merge," and "bob_replace." The configurations for these three usernames illustrate different configurations that correspond to the three different forms of the **access-profile** command. The three user configurations also illustrate setting up the autocommand for each form of the **access-profile** command.

Figure 5 shows the topology. The example following the figure shows a TACACS+ configuration file.

Figure 5 Example Topology for Double Authentication



TACACS+ Configuration File

This sample configuration shows authentication/authorization profiles on the TACACS+ server for the remote host "hostx" and for three users, with the usernames "bob_default," "bob_merge," and "bob_replace."

```

key = "mytacacskey"

default authorization = permit

#-----Remote Host (BRI)-----
#
# This allows the remote host to be authenticated by the local host
# during fist-stage authentication, and provides the remote host
# authorization profile.
#
#-----

user = hostx
{
    login = cleartext "welcome"
    chap = cleartext "welcome"

    service = ppp protocol = lcp {
        interface-config="ip unnumbered ethernet 0"
    }

    service = ppp protocol = ip {
        # It is important to have the hash sign and some string after
        # it. This indicates to the NAS that you have a per-user
        # config.

        inacl#3="permit tcp any 172.21.114.0 0.0.0.255 eq telnet"
        inacl#4="deny icmp any any"

        route#5="55.0.0.0 255.0.0.0"
        route#6="66.0.0.0 255.0.0.0"
    }

    service = ppp protocol = ipx {
        # see previous comment about the hash sign and string, in protocol = ip
        inacl#3="deny any"
    }

}

#----- "access-profile" default user "only acIs" -----
#
# - Without arguments, access-profile removes any access-lists it can find
# in the old configuration (both per-user and per-interface), and makes sure
# that the new profile contains ONLY access-list definitions.
#
#-----

user = bob_default
{
    login = cleartext "welcome"
    chap = cleartext "welcome"

    service = exec
    {

```

```

        # this is the autocommand that executes when bob_default logs in
        autocmd = "access-profile"
    }

    service = ppp protocol = ip {
        # Put whatever access-lists, static routes, whatever
        # here.
        # If you leave this blank, the user will have NO IP
        # access-lists (not even the ones installed prior to
        # this)!

        inacl#3="permit tcp any host 10.0.0.2 eq telnet"
        inacl#4="deny icmp any any"
    }

    service = ppp protocol = ipx {
        # Put whatever access-lists, static routes, whatever
        # here.
        # If you leave this blank, the user will have NO IPX
        # access-lists (not even the ones installed prior to
        # this)!
    }
}

#----- "access-profile merge" user -----
#
# With the 'merge' option, first all old access-lists are removed (as before),
# but then (almost) all AV pairs are uploaded and installed. This
# will allow for uploading any custom static routes, sap-filters, and so on,
# that the user may need in his or her profile. This needs to be used with
# care, as it leaves open the possibility of conflicting configurations.
#
#-----

user = bob_merge
{
    login = cleartext "welcome"
    chap = cleartext "welcome"

    service = exec
    {
        # this is the autocommand that executes when bob_merge logs in
        autocmd = "access-profile merge"
    }

    service = ppp protocol = ip
    {
        # Put whatever access-lists, static routes, whatever
        # here.
        # If you leave this blank, the user will have NO IP
        # access-lists (not even the ones installed prior to
        # this)!

        inacl#3="permit tcp any any"
        route#2="10.0.0.0 255.255.0.0"
        route#3="10.1.0.0 255.255.0.0"
        route#4="10.2.0.0 255.255.0.0"
    }
}

```

```

service = ppp protocol = ipx
{
    # Put whatever access-lists, static routes, whatever
    # here.
    # If you leave this blank, the user will have NO IPX
    # access-lists (not even the ones installed prior to
    # this)!
}

}

#----- "access-profile replace" user -----
#
#- With the 'replace' option,
# ALL old configuration is removed and ALL new configuration is installed.
#
# One caveat: access-profile checks the new configuration for address-pool and
# address AV pairs. As addresses cannot be renegotiated at this point, the
# command will fail (and complain) when it encounters such an AV pair.
# Such AV pairs are considered to be "invalid" for this context.
#-----

user = bob_replace
{
    login = cleartext "welcome"
    chap = cleartext "welcome"

    service = exec
    {
        # this is the autocommand that executes when bob_replace logs in
        autocmd = "access-profile replace"
    }

    service = ppp protocol = ip
    {
        # Put whatever access-lists, static routes, whatever
        # here.
        # If you leave this blank, the user will have NO IP
        # access-lists (not even the ones installed prior to
        # this)!

        inacl#3="permit tcp any any"
        inacl#4="permit icmp any any"

        route#2="10.10.0.0 255.255.0.0"
        route#3="10.11.0.0 255.255.0.0"
        route#4="10.12.0.0 255.255.0.0"
    }

    service = ppp protocol = ipx
    {
        # put whatever access-lists, static routes, whatever
        # here.
        # If you leave this blank, the user will have NO IPX
        # access-lists (not even the ones installed prior to
        # this)!
    }

}
#-----

```