



# Guard モジュールの設定

この章では、Cisco Anomaly Guard Module (Guard モジュール) のサービスの設定方法について説明します。

この章には、次の主要な項があります。

- [Guard モジュールのサービスのアクティブ化](#)
- [アクセス コントロールの設定：認証、認可、アカウンティングの使用](#)
- [Cisco Traffic Anomaly Detector Module との通信のイネーブル化](#)
- [日付と時刻の設定](#)
- [SSH 鍵の管理](#)
- [SFTP 接続の鍵の設定](#)
- [ホスト名の変更](#)
- [SNMP トラップのイネーブル化](#)
- [SNMP コミュニティ スtring の設定](#)

## Guard モジュールのサービスのアクティブ化

Guard モジュールでアクティブにするサービスを定義することができます。正しい機能をイネーブルにするためには、サービスをイネーブルにして、そのサービスへのアクセスを許可する必要があります。Guard モジュールのサービスのアクティベーションを制御し、特定の IP アドレスに対してアクセス権を付与または拒否することにより、Guard モジュールにアクセスし、制御する IP アドレスを制限することができます。

表 4-1 で、Guard モジュールのサービスについて説明します。

**表 4-1 Guard モジュール サービス**


| サービス                  | 説明  |
|-----------------------|---|
| <b>internode-comm</b> | ノード間通信サービス。Guard モジュールは、Cisco Traffic Anomaly Detector Module と通信チャネルを確立するときこのサービスを使用します。   |
| <b>snmp-server</b>    | <p>Simple Network Management Protocol (SNMP; 簡易ネットワーク管理プロトコル) サーバサービス。SNMP を使用して Guard モジュールにアクセスすることにより、Riverhead 管理情報ベース (Riverhead の専用 MIB、MIB2、および UC Davis の MIB) で定義された情報を取得することができます。</p> <p>MIB 定義の詳細については、このバージョンに付属の MIB ファイルを参照してください。</p> <p> (注) Riverhead MIB には、64 ビットのカウンタが含まれています。MIB を読み取るには、SNMP バージョン 2 をサポートするブラウザを使用する必要があります。</p> |
| <b>snmp-trap</b>      | SNMP トラップ サービス。snmp-trap サービスをアクティブにすると、Guard モジュールは SNMP トラップを生成します。詳細については、P.4-36 の「SNMP トラップのイネーブル化」を参照してください。   |

表 4-1 Guard モジュール サービス (続き)

| サービス | 説明  |
|------|---|
| ssh  | Secured Shell サービス (詳細については、 <a href="#">P.4-31</a> の「SSH 鍵の管理」を参照)。      |
| wbm  | Web-Based Management (WBM) サービス。Web ブラウザを使用して、Web 経由で Guard モジュールを制御できます。 |

Guard モジュールのサービスをアクティブにするには、次の手順を実行します。

**ステップ 1** Guard モジュールのサービスをイネーブルにします。次のコマンドを入力します。

```
service {internode-comm | snmp-server | snmp-trap | wbm}
```

Guard モジュールのサービスの説明については、[表 4-1](#) を参照してください。デフォルトでは、SSH 以外、Guard モジュールのすべてのサービスはディセーブルになっています。

**ステップ 2** Guard モジュールのサービスに対するアクセス権を付与し、接続を可能にします。次のコマンドを入力します。

```
permit service ip-address-general [ip-mask]
```

[表 4-2](#) に、**permit** コマンドの引数を示します。

表 4-2 permit コマンドの引数

| パラメータ                     | 説明  |
|---------------------------|---|
| <i>service</i>            | アクセスと操作の対象となるサービス。Guard モジュールのサービスの説明については、表 4-1 を参照してください。   |
| <i>ip-address-general</i> | アクセスを許可する IP アドレス。IP アドレスをドット区切り 10 進表記で入力します (たとえば 192.168.10.2)。すべての IP アドレスからのアクセスを許可するには、アスタリスク (*) を使用します。 |
| <i>ip-mask</i>            | (オプション) IP サブネット マスク。サブネット マスクをドット区切り 10 進表記で入力します (たとえば 255.255.255.0)。デフォルトのサブネット マスクは、255.255.255.255 です。    |



## 注意

セキュリティ上の理由から、すべての IP アドレスからのサービスへのアクセスを許可する (\* と入力する) ことは推奨しません。

## 例

```
user@GUARD-conf# service wbm
user@GUARD-conf# permit wbm 192.168.10.35
```

## アクセスコントロールの設定：認証、認可、アカウントिंगの使用

アクセスコントロールとは、Guard モジュールにアクセスできるユーザ、およびアクセス権を持ったユーザが使用できるサービスを制御することです。認証、認可、アカウントング（AAA）のセキュリティサービスは、アクセスを設定するための基本的なフレームワークとなります。

- 認証：ユーザにシステムおよびシステム サービスへのアクセスを許可する前に、そのユーザを識別する方法。
- 認可：システムへのアクセスを取得した後で、ユーザが実行することのできる内容を決定するプロセス。通常、このプロセスは、ユーザが認証され、システムの操作を開始した後で実行されます。
- アカウントング：ユーザが実行中または実行済みの内容を記録する処理。アカウントングにより、ユーザがアクセスしているサービスを追跡することができます。

Guard モジュールには、次のシステム ユーザ アカウントが事前設定されています。

- **admin**：admin ユーザ アカウントには、管理者アクセス権が設定されています。そのため、Guard モジュールの CLI とその機能にアクセスできます。Guard モジュールの CLI に初めて接続するときは、このアカウントのパスワードを設定する必要があります。新しいユーザ アカウントを設定するには admin ユーザ アカウントを使用します。
- **riverhead**：riverhead ユーザ アカウントには、ダイナミック（dynamic）のアクセス権が設定されています。Guard モジュールでは、このユーザ アカウントを使用して、Cisco Traffic Anomaly Detector Module と通信します。Guard モジュールの CLI に初めて接続するときは、このアカウントのパスワードを設定する必要があります。

Cisco Traffic Anomaly Detector Module は、riverhead ユーザ名を使用して Guard をリモートでアクティブにできます。

システム ユーザ アカウントは削除できません。

ユーザ定義を使用すると、Guard モジュールのユーザ コミュニティをドメインに分割し、安全な管理アクセスのために必要に応じてパスワードを割り当てることができます。初期設定が完了した後は、ユーザのアクションを監視できるように新しいアカウントを作成し、システム ユーザ アカウントは使用しないことをお勧めします。

次の各項では、アクセスコントロールの設定方法について説明します。

- [認証の設定](#)
- [認可の設定](#)
- [アカウントिंगの設定](#)
- [TACACS+ サーバのアトリビュートの設定](#)

## 認証の設定

ユーザが Guard モジュールにログインしようとするとき、または (**enable** コマンドを使用して) 上位の特権レベルを要求するときに、Guard モジュールで使用する認証方式を設定することができます。Guard モジュールは、次の認証オプションを提供します。

- **ローカル認証**：ローカル認証では、ローカルに設定されたログイン名およびイネーブルパスワードが認証に使用されます。これはデフォルトの認証方式です。詳細については、[P.4-8 の「ローカル認証の設定」](#)を参照してください。
- **TACACS+ 認証**：TACACS+ 認証では、1 つの TACACS+ サーバまたは複数の TACACS+ サーバのリストを使用してユーザが認証されます。  
ユーザを 1 つの TACACS+ サーバにしか設定しない場合、その TACACS+ サーバで当該ユーザに対して認可も設定する必要があります。設定しないと、そのユーザは **show** コマンドにしかアクセスできません。

シーケンシャルな認証リストを設定することができます。認証リストでは、ユーザの認証に使用する認証方式を定義します。この定義により、認証に使用する 1 つ以上の方式を指定することができます。したがって、最初の方式が失敗した場合は、認証のバックアップシステムが提供されます。

Guard モジュールは、最初にリストされた方式を使用してユーザを認証します。その方式が応答しない場合、Guard モジュールは 2 番目の認証方式を選択します。両方の認証方式を試してもうまくいかない場合、認証は失敗します。

分散認証方式を設定することもできます。Guard モジュールは、最初の TACACS+ サーバを使用してユーザを認証します。認証で拒否が返された場合、Guard モジュールは TACACS+ サーバリスト、および、存在する場合は代替の認証方式 (ローカル) をスキャンします。リストをすべて試してもうまくいかない場合、認証は失敗します。このオプションは、*first-hit* オプションを設定していない場合にのみ有効です。



(注) ユーザ データベースが、複数の TACACS+ サーバに分散している、または 1 つの TACACS+ サーバとローカル ユーザ データベースに分散している場合は、**no tacacs-server first-hit** コマンドを使用してください。

## 認証方式の設定

Guard モジュールで使用する認証方式を設定するには、次の手順を実行します。

**ステップ 1** TACACS+ 認証が必要な場合は、TACACS+ サーバ接続を設定します。詳細については、[P.4-18](#) の「[TACACS+ サーバのアトリビュートの設定](#)」を参照してください。

**ステップ 2** 認証方式を定義します。次のコマンドを入力します。

```
aaa authentication {enable | login} {local | tacacs+}
[tacacs+ | local]
```

[表 4-3](#) に、**aaa authentication** コマンドの引数を示します。

**表 4-3** aaa authentication コマンドのキーワード

| パラメータ                  | 説明                                      |
|------------------------|---|
| <b>enable</b>          | Guard モジュールは、上位の特権レベルに入るときに認証を行います。     |
| <b>login</b>           | Guard モジュールへのログイン時に認証が行われます。            |
| <b>local</b>           | Guard モジュールは、ローカル データベースを使用してユーザを認証します。 |
| <b>tacacs+</b>         | TACACS+ サーバによってユーザが認証されます。              |
| <b>tacacs+   local</b> | (オプション) 設定された方式が失敗した場合の代替認証方式を設定します。    |

コンソールセッションから Guard モジュールにアクセスした場合、Guard モジュールは、定義済みの認証方式に関係なく、ローカル ユーザ データベースを使用して認証します。

認証方式を変更するには、このコマンドを再入力します。

---

次の例は、上位の特権レベルに入る際に認証を行うように設定する方法を示しています。最初の認証方式は TACACS+ に設定され、2 番目の認証方式はローカル ユーザ データベースに設定されています。

```
user@GUARD-conf# aaa authentication enable tacacs+ local
```

## ローカル認証の設定

Guard モジュールには、管理者特権を持つユーザ名があらかじめ設定されています。このユーザ名を使用して新しいユーザを作成できます。ユーザ定義を使用すると、Guard モジュールのユーザ コミュニティをドメインに分割し、安全な管理アクセスのために必要に応じてパスワードを割り当てることができます。

1 つの TACACS+ サーバを使用した CLI ユーザの認証をイネーブルにするには、[P.4-6 の「認証の設定」](#)を参照してください。

## ユーザの追加

Guard モジュールのローカル データベースにユーザを追加するには、次のコマンドを入力します。

```
username username {admin | config | dynamic | show} [password]
```

[表 4-4](#) に、**username** コマンドの引数とキーワードを示します。



表 4-4 username コマンドの引数とキーワード

| パラメータ   | 説明   |
|---|--|
| <i>username</i>   | ユーザ名。1～63 文字の英数字の文字列です。大文字と小文字が区別され、先頭は英字である必要があります。この文字列にはスペースを含めることはできませんが、アンダースコアを含めることはできます。 |
| <b>admin</b>   <b>config</b>   <b>dynamic</b>   <b>show</b> | ユーザの特権レベル。詳細については、 <a href="#">表 3-1</a> を参照してください。  |
| <i>password</i>   | (オプション) パスワード。6～24 文字の文字列を入力します。スペースは使用できず、大文字と小文字が区別されます。パスワードを入力しない場合、入力するよう要求されます。            |

## 例

```
user@GUARD-conf# username Robbin config 1234
```

ユーザはパスワードをクリア テキストで入力しますが、Guard モジュールの設定ファイルでは、パスワードは暗号化されて表示されます。次に、Guard モジュールの設定ファイル (running-config) の表示例を示します。

```
username Richard config encrypted 840xdMk3
```

上の例の **encrypted** オプションは、パスワードが暗号化されていることを示しています。

Guard モジュールに設定されたユーザのリストを表示するには、**show running-config** コマンドまたは **show guard** コマンドを使用します。現在 CLI にログインしているユーザのリストを表示するには、**show users** コマンドを使用します。

Guard モジュールのユーザ リストからユーザを削除するには、**no username username** コマンドを使用します。

## パスワードの変更

ユーザは、自分自身のパスワードを変更することができます。管理者は、自分自身のパスワードと、他のすべてのユーザのパスワードを変更できます。

自分自身のパスワードを変更するには、次の手順を実行します。

---

**ステップ 1** 次のコマンドを入力します。

```
password
```

**ステップ 2** 現在のパスワードを入力します。新しいパスワードの入力を求めるプロンプトが表示されます。

**ステップ 3** 新しいパスワードを入力します。パスワードは、スペースを含まない、6 ~ 24 文字の英数字の文字列である必要があります。パスワードでは大文字と小文字が区別されます。新しいパスワードをもう一度入力し、確認するように求めるプロンプトが表示されます。

---

例

```
user@GUARD# password
Old Password: <old-password>
New Password: <new-password>
Retype New Password: <new-password>
```

管理者は、他のユーザのパスワードを変更できます。

特定のユーザのパスワードを変更するには、次の手順を実行します。

---

**ステップ 1** グローバル モードで次のコマンドを入力します。

```
password username-password
```

*username-password* 引数は、変更対象のパスワードを持つユーザです。

- ステップ 2** 新しいパスワードを入力します。パスワードは、スペースを含まない、6 ~ 24 文字の英数字の文字列である必要があります。パスワードでは大文字と小文字が区別されます。新しいパスワードをもう一度入力し、確認するように求めるプロンプトが表示されます。

---

次の例では、管理者はユーザ *John* のパスワードを変更しています。

```
user@GUARD# password Jose
New Password: <new-password>
Retype New Password: <new-password>
```

## 認可の設定

ユーザが使用できるサービスを制限することができます。認可がイネーブルの場合、Guard モジュールはユーザのプロファイルでそのユーザのアクセス権を確認します。プロファイルは、ローカル ユーザ データベースまたは TACACS+ セキュリティ サーバにあります。ユーザは、そのユーザのプロファイル内の情報で許可されている場合にのみ、要求したサービスへのアクセス権を付与されます。

ユーザがコマンドを実行しようとするときに Guard モジュールで使用する認可方式を設定することができます。Guard モジュールでは、次の認可オプションが提供されています。

- **TACACS+ 認可**：TACACS+ 認可では、TACACS+ サーバを使用してユーザが認可されます。後続のサーバが定義されている場合は、1 つのサーバとの通信が失敗した場合にのみ、そのサーバへのアクセスが開始されます。

TACACS+ 認可では、2 種類がサポートされています。実行認可では、ユーザが Guard モジュールへのログイン時に認証されるときに、一度ユーザの特権レベルが決定されます。コマンド認可では、ユーザがコマンドを入力すると、各コマンドの認可を取得するために TACACS+ サーバに対して確認が行われます。

TACACS+ 認可では、コマンドごとにアクセス権を指定することができます。

**注意**

**copy running-config** コマンドへの認可の付与には注意を払うことをお勧めします。

**copy running-config** コマンドの実行を認可すると、設定ファイル内ですべてのコマンドにそれぞれ認可を設定しているかどうかに関係なく、すべての設定コマンドに対して認可が与えられます。

- ローカル認可：ローカル認可では、コマンドグループのアクセスコントロールにローカルで設定されたユーザプロファイルが使用されます。許可は、指定された特権レベルのすべてのコマンドに対して定義されます。これはデフォルトの認可方式です。

シーケンシャルな認可リストを設定することができます。認可リストでは、ユーザの認可に使用する認可方式を定義します。この定義により、認可に使用する1つ以上の方式を指定することができます。したがって、最初の方式に対する通信が失敗した場合は、認可のバックアップシステムが提供されます。

Guard モジュールは、最初にリストされた方式を使用してユーザを認可します。その方式が応答しない場合、Guard モジュールは2番目の認可方式を選択します。両方の認可方式が成功しなかった場合、認可は失敗します。

Guard モジュールのローカル認可は、TACACS+ サーバへの通信に失敗した場合に実行することができます。

## ローカル許可の設定

Guard のサービスにアクセスできるかどうかは、ユーザの特権レベルによって決まります。システム管理者は、ユーザが使用できるサービスを制限することができます。Guard モジュールは、ユーザのプロファイルをチェックして、ユーザのアクセス権を確認します。認可されると、ユーザは、そのユーザのプロファイル内の情報で許可されている場合にのみ、要求したサービスへのアクセス権を付与されます。ユーザの特権レベルについては、表 3-1 を参照してください。

## パスワードを使用した特権レベルの割り当て

管理者は、ユーザの特権レベルへのアクセスを制限するパスワードを設定できません。

ローカルパスワードを設定して特権レベルへのアクセスを制御するには、次のコマンドを入力します。

```
enable password [level level] [password]
```

表 4-5 で、**enable password** コマンドの引数について説明します。

**表 4-5 enable password コマンドの引数**

| パラメータ           | 説明   |
|-----------------|--|
| <i>level</i>    | (オプション) 目的の特権レベル。このレベルには、 <b>admin</b> 、 <b>config</b> 、 <b>dynamic</b> 、 <b>show</b> のいずれかを指定できます。詳細については、表 3-1 を参照してください。デフォルトのレベルは <b>admin</b> です。 |
| <i>password</i> | (オプション) 特権レベルのパスワード。パスワードは、スペースを含まない、6 ~ 24 文字の英数字の文字列である必要があります。パスワードでは大文字と小文字が区別されます。パスワードを入力しない場合、入力するよう要求されます。                                     |

例

```
user@GUARD-conf# enable password level admin <password>
```

## ユーザ特権レベル間の移動

認可されたユーザは、ユーザ特権レベル間を移動することができます。

ユーザ特権レベル間を移動するには、次の手順を実行します。

**ステップ 1** 次のコマンドを入力します。

```
enable [level]
```

*level* 引数には、目的の特権レベルを指定します。このレベルには、**admin**、**config**、**dynamic** のいずれかを指定できます。デフォルトのレベルは **admin** です。詳細については、[表 3-1](#) を参照してください。

**ステップ 2** 特権レベルのパスワードを入力します。

例

```
user@GUARD> enable admin
Enter enable admin Password: <password>
```

下位の特権レベル (**show**) に戻る場合は、**disable** コマンドを使用します。

## 認可方式の設定

認可方式を設定するには、次の手順を実行します。

**ステップ 1** TACACS+ 認可が必要な場合は、TACACS+ サーバ接続を設定します。詳細については、[P.4-18](#) の「[TACACS+ サーバのアトリビュートの設定](#)」を参照してください。


**ステップ 2** 認可方式を定義します。次のいずれかのコマンドを入力します。

- **aaa authorization exec tacacs+**
- **aaa authorization commands level {local | tacacs+} [local]**

認可方式のシーケンシャルなリストを設定できます。各方式について、**aaa authorization** コマンドを入力します。認証方式を削除するには、このコマンドの **no** 形を使用します。

[表 4-6](#) に、**aaa authorization** コマンドの引数とキーワードを示します。

表 4-6 aaa authorization コマンドの引数とキーワード

| パラメータ           | 説明  |
|-----------------|---|
| <b>exec</b>     | <p>ユーザが EXEC シェルの実行を許可されているかどうかを判断するために認可が実行されます。Guard モジュールは、TACACS+ サーバに確認して、認証されたユーザの特権レベルを判断します。</p> <p> <b>注意</b> 認可を設定する前に、TACACS+ サーバにそのユーザを設定しておく必要があります。これを行わないと、Guard モジュールにアクセスできない場合があります。</p> |
| <b>commands</b> | 指定された特権レベルのすべてのコマンドに対して認可が実行されます。複数の特権レベルの認可を設定するには、認可が必要な特権レベルごとにこのコマンドを発行します。   |
| <b>level</b>    | 指定された特権レベルの認可を定義します。有効なエントリは、show、dynamic、config、および admin です。ユーザの特権レベルについては、表 3-1 を参照してください。   |
| <b>local</b>    | Guard モジュールのローカル データベースでユーザのアクセス権を確認します。  |
| <b>tacacs+</b>  | TACACS+ サーバでユーザのアクセス権を確認します。  |
| <b>local</b>    | (オプション) 設定された認可方式が失敗した場合の代替の認可方式を設定します。   |

パフォーマンスへの影響を考慮し、show 特権レベルのコマンドには認可を設定しないことをお勧めします。



(注) コンソールセッションから入力されたコマンドには、TACACS+ 認可は実行されません。

次の例は、*config* 特権レベルを必要とするコマンドの認可を設定する方法を示しています。最初の認可方式は TACACS+ に設定され、2 番目の認可方式はローカル ユーザ データベースに設定されています。

```
user@GUARD-conf# aaa authorization commands config tacacs+ local
```



### 注意

設定コマンド モードにアクセスできるようにするには、*dynamic* ユーザ特権レベルに対するアクセス権を付与するか、**configure** コマンドへのアクセス権を指定する必要があります。

## TACACS+ サーバの設定例

TACACS+ サーバのデータベースで、各コマンドの認可を指定することができます。

例

```
user=Zoe {
  cmd = protect {
    permit .*
  }
  cmd = "no protect" {
    permit .*
  }
  cmd = learning {
    deny policy*
  }
  cmd = "no learning" {
    deny .*
  }
  cmd = dynamic-filter {
    permit .*
  }
  cmd = "no dynamic-filter" {
    permit .*
  }
  cmd = flex-filter {
    deny .*
  }
  cmd = "no flex-filter" {
    deny .*
  }
}
```



## アカウントिंगの設定

アカウントING管理により、ユーザがアクセスしているサービスを追跡し、TACACS+ サーバにアカウントING情報を保存することができます。課金、レポート、またはセキュリティの目的のため、要求されたサービスのアカウントINGをイネーブルにします。デフォルトでは、Guard モジュールのアカウントING管理はディセーブルに設定されています。

アカウントINGを設定するには、次の手順を実行します。

**ステップ 1** TACACS+ サーバ接続を設定します。詳細については、[P.4-18](#) の「[TACACS+ サーバのアトリビュートの設定](#)」を参照してください。

**ステップ 2** 複数の特権レベルのアカウントINGを設定するには、アカウントINGが必要な特権レベルごとにこのコマンドを発行します。次のコマンドを入力します。

```
aaa accounting commands {show | dynamic | config | admin} stop-only  
{local | tacacs+}
```

[表 4-7](#) に `aaa accounting` コマンドのキーワードを示します。

**表 4-7** `aaa accounting` コマンドのキーワード

| パラメータ  | 説明   |
|--|--|
| <code>show   dynamic   config   admin</code> | 指定された特権レベルのアカウントINGを定義します（ユーザの特権レベルの詳細については、 <a href="#">表 3-1</a> を参照してください）。 |
| <code>stop-only</code>                       | コマンドの実行が終了したときにアクションを記録します。  |
| <code>tacacs+</code>                         | アカウントING情報の記録に TACACS+ サーバのデータベースを使用します。                                       |
| <code>local</code>                           | アカウントING情報を保存しません。   |

パフォーマンスへの影響を考慮し、アカウントING管理は `config` ユーザ特権レベルに対してのみイネーブルにすることをお勧めします。

アカウントिंग管理を削除するには、このコマンドの `no` 形を使用します。

---

次の例は、TACACS+ サーバ上で `config` 特権レベルを必要とするコマンドのアカウントिंगを設定する方法を示しています。

```
user@GUARD-conf# aaa accounting commands config stop-only tacacs+
```

## TACACS+ サーバの атрибуートの設定

TACACS+ サーバの атрибуートを設定するには、次の手順を実行します。



### 注意

TACACS+ 認証方式を適用する前に、TACACS+ サーバの атрибуートを設定しておく必要があります。これを行わないと、Guard モジュールにアクセスできない場合があります。

---

- ステップ 1 TACACS+ サーバの IP アドレスを設定します。詳細については、[P.4-19](#) の「[TACACS+ サーバの IP アドレスの設定](#)」を参照してください。
- ステップ 2 Guard モジュールが TACACS+ サーバへのアクセスに使用する暗号鍵を設定します。詳細については、[P.4-20](#) の「[TACACS+ サーバの暗号鍵の設定](#)」を参照してください。
- ステップ 3 (オプション) Guard モジュールが認証に使用する検索方式を設定します。詳細については、[P.4-20](#) の「[TACACS+ の検索の設定](#)」を参照してください。
- ステップ 4 (オプション) TACACS+ サーバの接続タイムアウトを設定します。詳細については、[P.4-21](#) の「[TACACS+ サーバの接続タイムアウトの設定](#)」を参照してください。

**ステップ 5** TACACS+ サーバ接続の統計情報を表示します。詳細については、[P.4-22](#) の「[TACACS+ サーバの統計情報の表示](#)」を参照してください。

Guard モジュールのユーザ特権レベルは、TACACS+ の特権番号に次のように対応しています。

- admin = 15
- config = 10
- dynamic = 5
- show = 0

## TACACS+ サーバの IP アドレスの設定

Guard モジュールが認証、認可、アカウントングに TACACS+ サーバのシークエンシャルなリストを使用するように設定できます。Guard モジュールは、リストされた TACACS+ サーバを使用して、ユーザを認証または認可したり、アカウントング イベントを送信したりします。そのサーバが応答しない場合、Guard モジュールは 2 番目のサーバを選択します。リストされたすべてのサーバを試してもうまくいかない場合、認証または認可は失敗します。

または、Guard モジュールがリストの最初の TACACS+ サーバだけを使用してユーザを認証するように設定することもできます（詳細については、[P.4-20](#) の「[TACACS+ の検索の設定](#)」を参照）。

リストには、各 TACACS+ サーバの IP アドレスを定義する必要があります。最大 9 つの TACACS+ サーバを定義できます。

リストに TACACS+ サーバを追加し、IP アドレスを割り当てるには、設定モードで次のコマンドを入力します。

```
tacacs-server host ip-address
```

*ip-address* 引数には、TACACS+ サーバの IP アドレスを指定します。

TACACS+ サーバは、入力した順序でリストに追加されます。リストには、最大 9 つのサーバを追加できます。

例

```
user@GUARD-conf# tacacs-server host 192.168.33.45
```

## TACACS+ サーバの暗号鍵の設定

TACACS+ サーバにアクセスするには、暗号鍵を設定する必要があります。コマンドの一部として入力される鍵は、TACACS+ サーバ上の鍵と一致している必要があります。鍵にスペースを含めることはできません。

サーバの暗号アクセス鍵を設定するには、設定モードで次のコマンドを入力します。

```
tacacs-server key tacacs-key
```

引数 *tacacs-key* は、英数字の文字列です。



(注)

---

定義できる暗号鍵は 1 つだけです。複数の TACACS+ サーバを使用する場合、Guard モジュールは同じ鍵を使用して、すべての TACACS+ サーバとの通信を暗号化します。

---

次の例は、TACACS+ サーバの暗号鍵を *TacacsKey* に設定する方法を示しています。

```
user@GUARD-conf# tacacs-server key <TacacsKey>
```

## TACACS+ の検索の設定

Guard モジュールが、1 つの認証拒否を最終的なものと見なし、他の TACACS+ サーバやローカル認証方式を使用したそれ以上の検索を中止するように設定することができます。この場合、**tacacs-server first-hit** コマンドを使用します。Guard モジュールは、サーバリストの最初の TACACS+ サーバだけを使用してユーザ認証を実行し、応答します。最初の TACACS+ サーバが応答しない場合、Guard はリストにある次のサーバを選択します。Guard は、ユーザ認証に対して最初に受け取る承認または拒否を最終的なものと見なし、他の TACACS+ サーバまたはローカルユーザデータベースを使用したそのユーザ認証の試行を停止します。

TACACS+ の検索方式を `first-hit` に設定しない場合、Guard モジュールはデフォルトでリスト内のすべての TACACS+ サーバでユーザを認証しようとします。ユーザ認証として `first-hit` 検索方式をイネーブルにする（`no tacacs-server first-hit` コマンドを使用する）場合、Guard は、リストの最初にある TACACS+ サーバを使用してユーザを認証します。最初のサーバが応答しなかった、またはユーザの認証に失敗した場合は、Guard はリストにある次のサーバを選択します。リストにあるすべての TACACS+ サーバが応答しなかった、またはユーザ認証に失敗した場合、ローカルの認証方式が設定されていないと、そのユーザ認証は失敗します。



---

(注) TACACS+ の検索方式は、認証にのみ適用されます。

---

Guard モジュールがリストの最初の TACACS+ サーバだけを使用してユーザを認証するように設定するには、設定モードで次のコマンドを入力します。

**tacacs-server first-hit**

例

```
user@GUARD-conf# tacacs-server first-hit
```

## TACACS+ サーバの接続タイムアウトの設定

Guard モジュールが TACACS+ サーバの応答を待つ時間を設定できます。タイムアウトが終了すると、Guard モジュールは次の TACACS+ サーバ（そのようなサーバが設定されている場合）との接続を確立しようとするか、ローカルの AAA にフォールバックします（そのようなフォールバックが設定されている場合）。フォールバックの方式が設定されていない場合、認証と認可は失敗します。



---

(注) すべての TACACS+ サーバとの通信に同じサーバタイムアウトが使用されます。

---

TACACS+ サーバの接続タイムアウトを設定するには、設定モードで次のコマンドを入力します。

**tacacs-server timeout *timeout***

*timeout* 引数には、Guard モジュールが TACACS+ サーバの応答を待つ時間（秒）を指定します。デフォルトのタイムアウトは 0 です。

例

```
user@GUARD-conf# tacacs-server timeout 600
```



ヒント

ネットワークに問題がある場合や、小さいタイムアウト値を使用していて、TACACS+ サーバの応答が遅いためにタイムアウトが継続的に発生する場合には、タイムアウトの値を大きくすることができます。

## TACACS+ サーバの統計情報の表示

TACACS+ サーバに関連する統計情報を表示することができます。統計データは、各サーバに対して提供されます。

TACACS+ 関連の統計情報を表示するには、**show tacacs statistics** コマンドを使用します。

TACACS+ の統計情報をクリアするには、**clear tacacs statistics** コマンドを使用します。

表 4-8 に、**show tacacs statistics** コマンド出力のフィールドを示します。

**表 4-8 show tacacs statistics コマンド出力のフィールドの説明**

| フィールド | 説明  |
|-------|---|
| PASS  | サービスが TACACS+ サーバに正常にアクセスし、アクセス権を付与された回数。 |
| FAIL  | サービスが TACACS+ サーバに正常にアクセスし、アクセス権を拒否された回数。 |
| ERROR | サービスが TACACS+ サーバにアクセスできなかった回数。           |

## Cisco Traffic Anomaly Detector Module との通信のイネーブル化

次のタスクをイネーブルにする場合は、Guard モジュールと Cisco Traffic Anomaly Detector Module (Detector モジュール) の間に安全な通信チャネルを確立できません。

- **Remote activation of zone protection**: Detector モジュールは、ゾーン トラフィックの異常を検出すると、通信チャネルを使用して Guard モジュールをアクティブにし、ゾーンを保護します。
- **Synchronization of zone configuration information**: Detector モジュールと Guard モジュールは、通信チャネルを介してゾーンの設定情報を交換します。

Guard モジュールは、次に示す 2 つのタイプの通信チャネルをサポートします。

- **Secure Sockets Layer (SSL)**: Remote activation of zone protection および Synchronization of zone configuration information をイネーブルにします。
- **セキュア シェル 2 (SSH2)**: Remote activation of zone protection のみをイネーブルにします。

Detector モジュールは、ゾーンを保護し、ゾーンの情報を同期させるためにアクティブにする Guard モジュールのリストを保持しています。このリストは、リモート Guard リストと呼ばれます。Detector モジュールは、リモート Guard リストに設定されている各 Guard モジュールと通信チャネルを確立します。

Detector モジュールは、SSL 通信チャネルを確立する前に、各 Guard モジュールと SSH 通信チャネルを確立します。したがって、両方のタイプのリストを設定済みである場合、SSH 通信チャネルの設定を行う必要はありません。SSH 通信チャネルは SSL 通信チャネルによって設定されるためです。

この項では、次のトピックについて取り上げます。

- [SSL 通信チャネルの設定](#)
- [SSH 通信チャネルの設定](#)

## SSL 通信チャネルの設定

Guard モジュールと Detector モジュールは、通信チャネルに SSL 接続を使用します。SSL は、プライバシー、認証、およびデータ整合性を組み合わせることにより、接続のセキュリティを確保します。SSL の高度なセキュリティは、デジタル証明書、秘密と公開の鍵交換ペア、および Diffie-Hellman 鍵合意パラメータによって実現されます。

SSL では、デバイス認証とデータ暗号化により、安全なネットワーク通信チャネルが提供されます。それぞれの Guard モジュールと Detector モジュールは、その通信チャネルを介してデバイスと通信を試みることで、デバイスを認証します。デバイス認証は、デジタル証明書とデバイス固有の情報（IP アドレスなど）を使用して実行されます。SSL は、Guard モジュールと Detector モジュールが交換するデータを暗号化します。データを解読できるのは、指定された受信者だけです。

安全な接続を確保するために、Detector モジュールは秘密鍵と公開鍵のペアを生成し、公開鍵をリモート Guard リスト内の Guard モジュールに配布します。

Guard モジュールで通信チャネルサービスをイネーブルにした後で、Detector モジュールから通信チャネルを確立します。Detector モジュールは最初に、ユーザー *riverhead* を使用して Guard モジュールに SSH2 通信を確立します。次に Detector モジュールは、安全な SSH2 接続を使用して、SSL 接続鍵を交換します。

この項では、次のトピックについて取り上げます。

- [SSL 通信チャネルの確立](#)
- [SSL 証明書の再生成](#)

## SSL 通信チャネルの確立

Guard モジュールと Detector モジュールとの間に SSL 通信チャネルを確立するには、次のタスクを実行する必要があります。

1. Guard モジュールと Detector モジュールの両方で、通信チャネルサービスをイネーブルにします。
2. Guard モジュールと Detector モジュールの両方で、その通信チャネル サービスへのアクセスを許可します。
3. Detector モジュールから通信チャネルを確立します。



Detector モジュールは最初に、ユーザ *riverhead* を使用して Guard モジュールに SSH2 通信を確立します。次に Detector モジュールは、安全な SSH2 接続を使用して、SSL 接続鍵を交換します。

**注意**

Guard モジュールで TACACS+ 認証を使用してユーザを認証している場合、Detector モジュールで SSH2 接続を確立するためには、TACACS+ サーバにユーザ *riverhead* を定義する必要があります。

Guard モジュールの通信チャネルをイネーブルにするには、Guard モジュール上で次の手順を実行します。

**(注)**

Detector モジュールの通信チャネルをイネーブルにする場合は、Detector モジュール上で同じコマンドを使用してください。

**ステップ 1** Detector モジュールの IP アドレスから Guard モジュール上の SSH サービスへのアクセスを許可するため、設定モードで **permit ssh ip-address-general [ip-mask]** コマンドを入力します。

引数 *ip-address-general* および *ip-mask* を使用すると、Guard モジュールへのアクセス権を付与する対象の Detector モジュールの IP アドレスを定義できます。



**(注)** SSH サービスはすでにイネーブルになっているので、ここでイネーブルにする必要はありません。

**ステップ 2** 通信チャネル サービスをイネーブルにするため、設定モードで **service internode-comm** コマンドを入力します。

- ステップ 3** Detector モジュールの IP アドレスから通信チャンネル サービスへのアクセスを許可するため、設定モードで **permit internode-comm ip-address-general [ip-mask]** コマンドを入力します。

引数 *ip-address-general* および *ip-mask* を使用すると、Guard モジュールへのアクセス権を付与する対象の Detector モジュールの IP アドレスを定義できます。



- (注) SSL 証明書内の Guard モジュールと Detector モジュールの ID は、IP アドレスに関連付けられています。通信チャンネルのそれぞれの端にある Guard モジュールまたは Detector モジュールのいずれかの IP アドレスを変更する場合、SSL 証明書を再生成する必要があります。P.4-26 の「SSL 証明書の再生成」を参照してください。

## SSL 証明書の再生成

SSL 証明書内の Guard モジュールと Detector モジュールを識別する鍵は、IP アドレスに関連付けられています。

次のいずれかの変更を行う場合、通信チャンネルの両端にある Guard モジュールと Detector モジュール用に新しい SSL 証明書を生成する必要があります。

- いずれか一方のデバイスの IP アドレスを変更する。
- いずれか一方のデバイスを交換（スワップアウト）する。

新しい SSL 証明書を生成するには、まず、現在使用している証明書を両方のデバイスで削除する必要があります。

現在使用している SSL 証明書を消去するには、次の手順を実行します。

- ステップ 1** Guard モジュールの SSL 証明書を Detector モジュールから削除します。

- ステップ 2** Detector モジュールの SSL 証明書を Guard モジュールから削除します。

設定モードで次のコマンドを入力します。

```
cert remove cert-host-ip
```

*cert-host-ip* 引数には、Detector モジュールの IP アドレスを指定します。Guard モジュールと通信チャネルを確立したすべての Detector モジュールの SSL 証明書を削除する場合は、アスタリスク (\*) を入力します。

**ステップ 3** Guard モジュールのデバイスを物理的に交換した場合は、Guard モジュールの SSH ホスト鍵を Detector モジュールから削除する必要があります。

**ステップ 4** Guard モジュールと Detector モジュールの間に新しい SSL 通信チャネルを確立します。詳細については、[P.4-24](#) の「[SSL 通信チャネルの確立](#)」を参照してください。

---

次の例は、SSL 証明書を削除する方法を示しています。

```
user@GUARD-conf# cert remove 10.56.36.4
```

## SSH 通信チャネルの設定

トラフィックの異常を検出すると、Detector モジュールはそのイベントをログに記録するか、SSH 接続を使用して Guard モジュールをアクティブにしてゾーンを保護します。SSH 通信チャネルを使用している場合、Detector モジュールは次のタスクを実行できません。

- ゾーン設定を同期化する。
- Guard モジュールを監視して、ゾーンへの攻撃が終了したことを確認する。したがって、Detector モジュールを検出およびラーニングの動作状態でアクティブにしても、Detector モジュールはゾーンへの攻撃が終了したことを確認できないため、リモート Guard をアクティブにした後でゾーントラフィックのラーニングを続行しません。

安全な SSH2 接続を確保するために、Detector モジュールは秘密 SSH 鍵と公開 SSH 鍵のペアを生成し、公開 SSH 鍵を Detector モジュールのリモート Guard リスト内の Guard モジュールに配布します。

この項では、次のトピックについて取り上げます。

- [SSH 通信チャネルの確立](#)
- [SSH 通信チャネル鍵の再生成](#)

## SSH 通信チャネルの確立

Guard モジュールと Detector モジュールとの間に通信チャネルを確立するには、次のタスクを実行する必要があります。

1. Detector モジュールの IP アドレスから Guard モジュール上の SSH サービスへのアクセスを許可するため、**permit ssh** コマンドを入力します。
2. Detector モジュールから SSH 通信チャネルを確立します。



### 注意

Guard で TACACS+ 認証を使用してユーザを認証している場合、Detector モジュールで SSH2 接続を確立するためには、TACACS+ サーバにユーザ *riverhead* を定義する必要があります。

Guard モジュールのデバイスを交換（スワップアウト）した場合は、SSH 通信チャネルを再生成する必要があります。[P.4-28](#) の「[SSH 通信チャネル鍵の再生成](#)」を参照してください。

## SSH 通信チャネル鍵の再生成

Guard モジュールのデバイスを物理的に交換する場合、次の手順を実行して新しい SSH 通信チャネルを確立します。

- ステップ 1** SSH2 ホスト鍵を Detector モジュールから削除します。設定モードにおいて、Detector モジュール上で **no host-keys ip-address-general** コマンドを入力します。

*ip-address* 引数には、リモートデバイスの IP アドレスを指定します。

Guard モジュールにリストされているホスト鍵を表示するには、**show host-keys** コマンドを使用します。

**ステップ 2** 次のいずれかのアクションを実行します。

- Detector モジュールから新しい SSH 通信チャネルを確立する (P.4-28 の「SSH 通信チャネルの確立」を参照)。  
または
  - Detector モジュールの公開鍵をリモート Guard に手動で追加する。  
Detector モジュールの公開 SSH 鍵を表示するには、Detector モジュール上で **show public-key** コマンドを使用します。  
Detector モジュールの公開 SSH 鍵を設定するには、Guard モジュール上で **key add** コマンドを使用します。
-

## 日付と時刻の設定

時刻と日付を設定するには、設定モードで次のコマンドを入力します。

```
date MMDDhhmm[[CC]YY][.ss]
```

表 4-9 に、**date** コマンドの引数とキーワードを示します。

**表 4-9**      **date コマンドの引数**

| パラメータ      | 説明                                     |
|------------|--|
| <i>MM</i>  | 数字で表した月。                               |
| <i>DD</i>  | 月の日付。                                  |
| <i>hh</i>  | 24 時間表記の時間。                            |
| <i>mm</i>  | 分。                                     |
| <i>CC</i>  | (オプション) 年の最初の 2 桁 (たとえば <b>2005</b> )。 |
| <i>YY</i>  | (オプション) 年の最後の 2 桁 (たとえば <b>2005</b> )。 |
| <i>.ss</i> | (オプション) 秒 (小数点が必要)。                    |

次の例は、日付を 2003 年 10 月 8 日に設定し、時刻を午後 5 時 10 分 (17 時 10 分) 17 秒に設定する方法を示しています。

```
user@GUARD-conf# date 1008171003.17
Wed Oct  8 17:10:17 EDT 2003
```

## SSH 鍵の管理

Guard モジュールは、安全なリモート ログインのために SSH をサポートします。SSH 鍵のリストを追加すると、ログイン名とパスワードを入力せずに、リモートデバイスから Guard モジュールへの安全な通信をイネーブルにできます。

次の各項では、Guard モジュールの SSH 鍵リストの操作方法について説明します。

- [SSH 鍵の追加](#)
- [SSH 鍵の削除](#)

### SSH 鍵の追加

ログイン名とパスワードを入力せずに SSH 接続をイネーブルにするには、Guard モジュールの SSH 鍵リストにリモート接続の SSH 公開鍵を追加します。

設定モードで次のコマンドを入力します。

```
key add [user-name] {ssh-dsa | ssh-rsa} key-string comment
```

表 4-10 に、`key add` コマンドの引数とキーワードを示します。

**表 4-10 key add コマンドの引数とキーワード**

| パラメータ             | 説明   |
|-------------------|--|
| <i>user-name</i>  | (オプション) 指定されたユーザの SSH 鍵を追加します。他のユーザの SSH 鍵を追加できるのは管理者だけです。<br>デフォルトは、現行ユーザの SSH 鍵の追加です。  |
| <b>ssh-dsa</b>    | SSH2-DSA 鍵のタイプ。  |
| <b>ssh-rsa</b>    | SSH2-RSA 鍵のタイプ。  |
| <i>key-string</i> | Detector またはリモート端末で作成された公開 SSH 鍵。鍵ストリングは、8,192 ビットまでに制限されています。<br>鍵タイプの識別 ( <code>ssh-rsa</code> または <code>ssh-dsa</code> ) を除いた完全な鍵をコピーする必要があります。 |

表 4-10 key add コマンドの引数とキーワード（続き）

| パラメータ          | 説明  |
|----------------|---|
| <i>comment</i> | デバイスの説明。コメントの形式は、通常、鍵の生成に使用されるユーザとマシンを表す <code>user@hostname</code> になります。たとえば、Detector で生成される SSH 公開鍵に使用されるデフォルトのコメントは、 <b>root@DETECTOR</b> です。 |

例

```
user@GUARD-conf# key add ssh-rsa 14513797528175730. . .user@Guard
module.com
```

## SSH 鍵の削除

リストから SSH 鍵を削除できます。SSH 鍵を削除すると、次に Guard モジュールと SSH セッションを確立するときには認証を受ける必要があります。

Guard モジュールから SSH 鍵を削除するには、設定モードで次のコマンドを入力します。

```
key remove [user-name] key-string
```

表 4-11 で、`key remove` コマンドの引数について説明します。

表 4-11 key remove コマンドの引数

| パラメータ             | 説明  |
|-------------------|---|
| <i>user-name</i>  | (オプション) 指定のユーザの SSH 鍵を削除します。<br>他のユーザの SSH 鍵を削除できるのは管理者だけです。デフォルトは、現行ユーザの SSH 鍵の削除です。     |
| <i>key-string</i> | 削除する公開 SSH 鍵。<br>プロンプトに SSH 公開鍵をペーストします。識別フィールド (ssh-rsa または ssh-dsa) は除き、鍵だけをペーストしてください。 |



次の例は、`key remove` コマンド用にカットアンドペーストを行えるように、ユーザ鍵を表示する方法を示しています。

```
user@GUARD-conf# show keys Lilac
ssh-rsa 2352345234523456... user@Guard module.com
user@GUARD-conf# key remove Lilac 2352345234523456...
```

## SFTP 接続の鍵の設定

Guard モジュールは、SSH2 の上層に位置するセキュア FTP (SFTP) をサポートします。SFTP では、公開鍵による認証と強力なデータ暗号化を使用しています。これにより、ログイン、データ、およびセッションの情報が送信中に傍受されたり変更されたりすることを防止できます。

SFTP サーバに公開鍵を設定するには、Guard モジュール上で次のアクションを実行します。

---

**ステップ 1** Guard モジュールの公開鍵を表示します。設定モードで `show public-key` コマンドを使用します。

鍵が存在する場合は、[ステップ 2](#) を省略し、[ステップ 3](#) に進みます。

鍵が存在しない場合は、[ステップ 2](#) に進みます。

**ステップ 2** 秘密および公開鍵ペアを生成します。設定モードで次のコマンドを入力します。

```
key generate
```

SSH の鍵ペアがすでに存在している場合は、次のメッセージが表示されます。

```
/root/.ssh/id_rsa already exists.
Overwrite (y/n)?
```

目的のオプションを入力します。

Guard モジュールが、公開および秘密鍵ペアを作成します。Guard モジュールの公開鍵を表示するには、設定モードで `show public-key` コマンドを使用します。

**ステップ 3** 公開鍵をコピーし、SFTP サーバにある鍵のファイルにペーストします。

たとえば、Linux オペレーティング システムに設置された SFTP サーバに root ユーザで接続する場合は、Guard モジュールの公開鍵を `/root/.ssh/authorized_keys2` ファイルに追加します。

鍵は 1 行に収まるようにコピーしてください。鍵が 2 行としてコピーされた場合は、1 行目の末尾にある改行文字を削除します。

---

## ホスト名の変更

Guard モジュールのホスト名を変更できます。この変更はすぐに反映され、新しいホスト名は自動的に CLI プロンプト スtring に組み込まれます。

Guard モジュールのホスト名を変更するには、設定モードで次のコマンドを入力します。

**hostname** *name*

*name* 引数には、新しいホスト名を指定します。

例

```
user@GUARD-conf# hostname CiscoGuard  
admin@CiscoGuard-conf#
```

## SNMP トラップのイネーブル化

Guard モジュールが SNMP トラップを送信し、Guard モジュールで発生する重大なイベントを管理者に通知するように設定できます。Guard モジュールの SNMP トラップ ジェネレータを設定して、トラップ情報のスコープを定義することができます。

トラップのログは、Guard モジュールのイベント ログに記録され、トラップ条件が発生すると、SNMP エージェントがトラップを送信するかどうかに関係なく、イベント モニタに表示されます。

Guard モジュールが SNMP トラップを送信するように設定するには、次の手順を実行します。

- ステップ 1** SNMP トラップ ジェネレータ サービスをイネーブルにします。設定モードで次のコマンドを入力します。

```
service snmp-trap
```

- ステップ 2** SNMP トラップ ジェネレータ パラメータ（トラップの宛先 IP アドレスとトラップ情報のスコープ）を設定します。次のコマンドを入力します。

```
snmp trap-dest ip-address [community-string [min-severity]]
```

表 4-12 に、`snmp trap-dest` コマンドの引数を示します。

**表 4-12 snmp trap-dest コマンドの引数**

| パラメータ                         | 説明  |
|-------------------------------|---|
| <code>ip-address</code>       | 宛先ホストの IP アドレス。   |
| <code>community-string</code> | (オプション) トラップとともに送信されるコミュニティ ストリング。このストリングは、宛先ホスト用に定義されたコミュニティ ストリングと一致する必要があります。デフォルトのコミュニティ ストリングは、 <code>public</code> です。1 ~ 15 文字の英数字の文字列を入力します。この文字列にスペースを含めることはできません。 |

表 4-12 snmp trap-dest コマンドの引数（続き）

| パラメータ               | 説明  |
|---------------------|---|
| <i>min-severity</i> | <p>(オプション) トラップ情報のスコープ。重大度レベルの範囲の下限を指定してスコープを定義します。この定義により、トラップは指定された重大度レベル以上のすべてのイベントを表示します。たとえば、重大度レベル <i>Warnings</i> を指定すると、トラップは <i>Warnings</i> から <i>Emergencies</i> までのすべての重大度レベルのイベントを表示します。重大度レベルのオプションを次に示します。</p> <ul style="list-style-type: none"> <li>• <b>Emergencies</b> : システムは使用不能 (重大度 = 0)</li> <li>• <b>Alerts</b> : 即時のアクションが必要 (重大度 = 1)</li> <li>• <b>Critical</b> : 危険な状態 (重大度 = 2)</li> <li>• <b>Errors</b> : エラー状態 (重大度 = 3)</li> <li>• <b>Warnings</b> : 警告状態 (重大度 = 4)</li> <li>• <b>Notifications</b> : 正常ではあるが、重要な状態 (重大度 = 5)</li> <li>• <b>Informational</b> : 情報通知のためのメッセージ (重大度 = 6)</li> <li>• <b>Debugging</b> : デバッグ メッセージ (重大度 = 7)</li> </ul> <p>デフォルトでは、レポートにはすべての重大度レベルのイベントが表示されます。</p> |

SNMP トラップ ジェネレータ パラメータを削除するには、**no snmp trap-dest** コマンドを使用します。すべての SNMP トラップ宛先パラメータを削除するには **\*** を使用します。

次に、*errors* 以上の重大度レベルのトラップが、SNMP コミュニティ スtring *tempo* とともに宛先 IP アドレス *192.168.100.52* に送信される例を示します。

```
user@GUARD-conf# snmp trap-dest 192.168.100.52 tempo errors
```

## SNMP コミュニティ スtring の設定

Guard モジュールの SNMP サーバにアクセスし、管理情報ベース 2 (MIB2) および Riverhead の専用 MIB で定義された情報を取得することができます。コミュニティ スtring はパスワードのように機能し、Guard モジュールの SNMP エージェントからの読み取りアクセスを許可します。Guard モジュールの SNMP コミュニティ スtring を設定して、異なる組織のクライアントが異なるコミュニティ スtring を使用して SNMP エージェントにアクセスできるようにすることができます。

SNMP コミュニティ スtring を追加するには、設定モードで次のコマンドを入力します。

**snmp community community-string**

*community-string* 引数には、目的の Guard モジュールのコミュニティ スtring を指定します。1 ~ 15 文字の英数字の文字列を入力します。この文字列にスペースを含めることはできません。Guard モジュールのデフォルトのコミュニティ スtring は *riverhead* です。コミュニティ名はいくつでも指定できます。コミュニティ スtring を削除するには、**no community string** コマンドを使用します。すべての SNMP コミュニティ スtring を削除するには \* を使用します。

例

```
user@GUARD-conf# snmp community tempo
```