セキュア シェルの実装

セキュア シェル（SSH）は、Berkeley の r ツールへのセキュアな置換を提供するアプリケーションおよびプロトコルです。プロトコルは標準の暗号メカニズムを使用してセッションの安全を確保します。アプリケーションは Berkeley の **rexec** および **rsh** ツールと同様に使用できます。

SSH サーバとして、SSH バージョン 1（SSHv1）と SSH バージョン 2（SSHv2）の 2 種類のバージョンを使用できます。SSHv1 は Rivest、Shamir、Adelman（RSA）キーを使用し、SSHv2 はデジタル署名アルゴリズム（DSA）キーまたは Rivest、Shamir、Adelman（RSA）キー、または楕円曲線デジタル署名アルゴリズム（ECDSA）キーを使用します。Cisco ソフトウェアは SSHv1 と SSHv2 の両方をサポートしています。

このモジュールでは、セキュア シェルの実装方法について説明します。

セキュア シェルの実装

セキュア シェル（SSH）は、Berkeley の r ツールへのセキュアな置換を提供するアプリケーションおよびプロトコルです。プロトコルは標準の暗号メカニズムを使用してセッションの安全を確保します。アプリケーションは Berkeley の **rexec** および **rsh** ツールと同様に使用できます。

SSH サーバとして、SSH バージョン 1（SSHv1）と SSH バージョン 2（SSHv2）の 2 種類のバージョンを使用できます。SSHv1 は Rivest、Shamir、Adelman（RSA）キーを使用し、SSHv2 はデジタル署名アルゴリズム（DSA）キーまたは Rivest、Shamir、Adelman（RSA）キー、または楕円曲線デジタル署名アルゴリズム（ECDSA）キーを使用します。Cisco ソフトウェアは SSHv1 と SSHv2 の両方をサポートしています。

このモジュールでは、セキュア シェルの実装方法について説明します。

セキュアシェルの実装に関する前提条件

セキュア シェルを実装するには、次の前提条件を満たす必要があります。

- 必要なイメージをルータにダウンロードします。SSH サーバと SSH クライアントでは、暗号化パッケージ（データ暗号規格（DES）、トリプルDES、およびAES）をシスコからご使用のルータにダウンロードする必要があります。

（注） Cisco IOS XR ソフトウェア Release 7.0.1 以降では、SSH および SFTP コンポーネントはベースライン Cisco IOS XR ソフトウェア イメージ自体で使用できます。詳細については、「ベースライン Cisco IOS XR ソフトウェアイメージの SSH および SFTP （2 ページ）」を参照してください。

- ローカル アクセスまたはリモート アクセス用にユーザ認証を設定します。認証、許可、アカウンティング（AAA）の有無に関係なく、認証を設定できます。
- AAA 認証および認可はセキュア ファイル転送プロトコル（SFTP）で機能するように正しく設定されている必要があります。

ベースライン Cisco IOS XR ソフトウェアイメージの SSH および SFTP

Cisco IOS XR ソフトウェア Release 7.0.1 以降では、Cisco IOS XR セキュリティパッケージ（k9sec パッケージ）に含まれていた管理プレーンとコントロールプレーンのコンポーネントがベース Cisco IOS XR ソフトウェア イメージに移動されています。これには、SSH、SCP、SFTP が含まれます。ただし、データ プレーン コンポーネント（Dot1x など）は、輸出コンプライアンス規制に従ってセキュリティパッケージの一部として保持されます。このパッケージコンポーネントの分離により、ソフトウェアのモジュール化が進みます。また、要件に従って柔軟にセキュリティパッケージを含めたり除外したりできます。

コントロールプレーンがFIPS認定アルゴリズムをネゴシエートできるように、ベースパッケージとセキュリティパッケージは FIPS に対応しています。

セキュア シェルの実装に関する制約事項

SSH の基本的な制約事項と SFTP 機能の制限は、次のとおりです。

- 外部クライアントがルータに接続するには、ルータにRSA（SSHv1またはSSHv2の場合）またはDSA（SSHv2の場合）またはECDSA（SSHv2の場合）キー ペアが設定されている必要があります。ルータから外部ルーティング デバイスに SSH クライアント接続を開始する場合、ECDSA、DSA および RSA キーは必要ありません。これは SFTP も同様です。SFTP はクライアント モードでのみ動作するため、ECDSA、DSA および RSA キーは必要ありません。
- SFTP が正常に動作するには、リモート SSH サーバは SFTP サーバ機能をイネーブルにする必要があります。たとえば、**/etc/ssh2/sshd2_config** などの行を使用して、SFTP サブシステムを処理するように SSHv2 サーバを設定します。
- **subsystem-sftp /usr/local/sbin/sftp-server**

- SFTP サーバは通常パブリック ドメインの SSH パッケージの一部として含まれており、デフォルトの構成では有効になっています。
- SFTP は、SFTP サーバ バージョン OpenSSH_2.9.9p2 以上と互換性があります。
- SSH サーバおよび SFTP サーバでは、RSA ベースのユーザ認証がサポートされています。ただし、SSH クライアントではこの認証はサポートされていません。
- サポートされるアプリケーションは、実行シェルおよび SFTP のみです。
- SFTP クライアントは、ワイルドカード（* ?、[]）を含むリモート ファイル名をサポートしません。ソース ファイルをルータにダウンロードするには、ユーザは **sftp** コマンドを複数回発行するか、リモート ホストからすべてのソース ファイルを表示する必要があります。アップロードについては、この項の 1 番目から 3 番目までの箇条書きで示した問題が解決されている場合、ルータ SFTP クライアントはワイルドカードを使用した複数ファイルの指定をサポートできます。
- SSH サーバの暗号化設定は、AES128、AES192、AES256、トリプル DES の順です。サポートされていない暗号の場合、サーバはクライアントの要求をすべて拒否し、SSH セッションは続行されません。
- vt100 以外の端末タイプの使用はサポートされていません。この場合、ソフトウェアは警告メッセージを生成します。
- SSH クライアントでは、パスワード メッセージとして「none」を使用することはサポートされていません。
- ルータ インフラストラクチャは UNIX 同様のファイル権限をサポートしていないため、ローカル デバイスに作成されたファイルは元の権限情報を失います。リモート ファイル システム上に作成されたファイルの場合、ファイル権限は宛先ホストの umask に従い、変更時間および最終アクセス時間はコピーの時間になります。

SSH の設定

SSH を設定するには、次のタスクを実行します。

（注） SSHv1 の設定では、ステップが 1 ～ 4 が必須です。SSHv2 の設定では、ステップ ～ステップ 4 はオプションです。

手順

ステップ 1 **configure**

ステップ 2 **hostname** *hostname*

例：

```
RP/0/RP0/cpu 0: router(config)# hostname router1
```

ルータのホスト名を設定します。

ステップ 3 **domain name** *domain-name*

例：

```
RP/0/RP0/cpu 0: router(config)# domain name cisco.com
```

このソフトウェアで使用するデフォルトのドメイン名を定義し、不完全なホスト名のドメインを補完します。

ステップ 4 **commit**

ステップ 5 **crypto key generate rsa** [**usage keys** | **general-keys**] [*keypair-label*]

例：

```
RP/0/RP0/cpu 0: router# crypto key generate rsa general-keys
```

RSA キー ペアを生成します。RSA キー モジュラスの範囲は 512 ～ 4096 ビットです。

- RSA キー ペアを削除するには、**crypto key zeroize rsa** コマンドを使用します。
- このコマンドは SSHv1 だけに使用されます。

ステップ 6 crypto key generate dsa

例：

```
RP/0/RP0/cpu 0: router# crypto key generate dsa
```

ルータでローカルおよびリモート認証を行う SSH サーバをイネーブルにします。サポートされているキーのサイズは 512、768 および 1024 ビットです。

- 推奨する最小絶対サイズは 1024 ビットです。
- DSA キー ペアを生成します。

  DSA キー ペアを削除するには、**crypto key zeroize dsa** コマンドを使用します。
- このコマンドは、SSHv2 だけに使用されます。

ステップ 7 **crypto key generate ecdsa** [**nistp256** | **nistp384** |**nistp521**]

例：

```
RP/0/RP0/cpu 0: router# crypto key generate ecdsa nistp256
```

ECDSA キーペアを生成します。サポートされている ECDSA 曲線タイプは、Nistp256、Nistp384 および Nistp521 です。

- ECDSA キー ペアを削除するには、**crypto key zeroize ecdsa [nistp256 | nistp384 | nistp521]** コマンドを使用します。

- このコマンドは SSHv2 だけに使用されます。

ステップ 8 configure

例：

```
RP/0/RP0/cpu 0: router# configure
```

モードを開始します。

ステップ 9 **ssh timeout** *seconds*

例：

```
RP/0/RP0/cpu 0: router(config)# ssh timeout 60
```

（任意）AAA へのユーザ認証に対するタイムアウト値を設定します。

- 設定された時間内にユーザ自身の認証が AAA に対してできないと、接続は中断されます。
- 値を設定しなければ、30 秒のデフォルト値が使用されます。範囲は 5 ～ 120 です。

ステップ 10 次のいずれかを実行します。

- **ssh server** [**vrf** *vrf-name*]
- ssh server v2

例：

```
RP/0/RP0/cpu 0: router(config)# ssh server v2
```

- （任意）32 文字までの指定された VRF を使用して SSH サーバを起動します。VRF が指定されていない場合、デフォルトの VRF が使用されます。

  SSH サーバが指定された VRF の接続をこれ以上受信しないようにするには、このコマンドの **no** 形式を使用します。VRF が指定されていない場合、デフォルトが使用されます。

（注） SSH サーバは複数の VRF を使用するように設定できます。

- （任意）**ssh server v2** コマンドを使用して SSHv2 オプションを設定すると、SSH サーバは SSHv2 クライアントだけを受け入れるようになります。**ssh server v2** コマンドを選択すると、SSH v2 クライアント接続だけが許可されます。

ステップ 11 **commit**

ステップ 12 show ssh

例：

```
RP/0/RP0/cpu 0: router# show ssh
```

（任意）ルータへの着信および発信の SSHv1 接続と SSHv2 接続をすべて表示します。

ステップ 13 show ssh session details

例：

```
RP/0/RP0/cpu 0: router# show ssh session details
```

（任意）ルータに対する SSHv2 接続の詳細レポートを表示します。

ステップ 14 **show ssh history**

例：

```
RP/0/RP0/cpu 0: router# show ssh history
```

（オプション）終了した最後の 100 個の SSH 接続を表示します。

ステップ 15 **show ssh history details**

例：

```
RP/0/RP0/cpu 0: router# show ssh history details
```

（オプション）終了した最後の 100 個の SSH 接続を詳細情報とともに表示します。このコマンドは **show ssh session details** コマンドに似ていますが、セッションの開始時刻と終了時刻も示されます。

ステップ 16 **show tech-support ssh**

例：

```
RP/0/RP0/cpu 0: router# show tech-support ssh
```

（オプション）システム情報を表示する `show` コマンドを自動的に実行します。

（注） SSH 接続のネゴシエーションを行う際の優先順位は、次のとおりです。

1. ecdsa-nistp-521
2. ecdsa-nistp-384
3. ecdsa-nistp-256
4. rsa
5. dsa

SSH ホストキーペアの自動生成

この機能により、DSA、ECDSA（**ecdsa-nistp256**、**ecdsa-nistp384**、**ecdsa-nistp521** など）および RSA アルゴリズムの SSH ホストキーペアを自動的に生成できます。そのため、ルータの起動後に各 SSH ホストキーペアを明示的に生成する必要がなくなります。キーはすでにシステムに存在しているため、SSH クライアントは、基本的な SSH 設定を使用してルータが起動し

た直後に SSH サーバとの接続を確立できます。これは特に、ゼロタッチプロビジョニング（ZTP）およびゴールデン ISO の起動シナリオで役立ちます。

このように自動化される前は、**crypto key generate** コマンドを実行して必要なホストキーペアを生成する必要がありました。

この機能が導入されたためホストキーペアは自動生成されますが、従来どおり SSH サーバで必要なアルゴリズムだけを柔軟に選択できます。このためには、XR コンフィギュレーションモードで **ssh server algorithms host-key** コマンドを使用します。または、XR EXEC モードで既存の **crypto key zeroize** コマンドを使用して、不要なアルゴリズムを削除することもできます。

この機能が導入される前は、XR EXEC モードで **crypto key generate** コマンドを実行して必要なホストキーペアを生成する必要がありました。

（注） バージョン 1 からバージョン 2 へのシステム アップグレード シナリオでは、すでにバージョン 1 で生成されている SSH ホストキーペアは自動的に生成されません。ホストキーペアは、バージョン 1 で生成されなかった場合にのみ自動的に生成されます。

SSH クライアントの設定

SSH クライアントを設定するには、次の作業を実行します。

手順

ステップ 1 **configure**

ステップ 2 **ssh client knownhost** *device* **:**/*filename*

例：

```
RP/0/RP0/cpu 0: router(config)# ssh client knownhost slot1:/server_pubkey
```

（任意）この機能がクライアント側でサーバの公開キー（pubkey）を認証し、確認できるようにします。

（注） ファイル名の完全なパスが必要です。コロン（:）とスラッシュ（/）も必要です。

ステップ 3 **commit**

ステップ 4 **ssh** {*ipv4-address* | *ipv6-address* | *hostname*} [**username** *user-* **cipher** | **source-interface** *type instance*]

発信 SSH 接続をイネーブルにします。

- SSHv2 サーバを実行するには、VRF が必要です。これはデフォルトの VRF でも固有の VRF でも構いません。VRF に関する変更は SSH v2 サーバのみに適用されます。

- SSH クライアントにより、リモート ピアへの SSHv2 接続が試みられます。リモート ピアで SSHv1 サーバしかサポートされていない場合、そのピアでリモート サーバへの SSHv1 接続が内部生成されます。
- **cipher des** オプションは、SSHv1 クライアントでのみ使用可能です。
- SSHv1 クライアントは、3DES 暗号化アルゴリズム オプションだけをサポートします。このオプションは、これらの SSH クライアントに対してだけ、まだデフォルトで使用可能です。
- *hostname* 引数が使用され、ホストに IPv6 と IPv4 の両方のアドレスがある場合、IPv6 アドレスが使用されます。

- SSHv1 を使用しており、SSH 接続が拒否されている場合は、RSA キーペアがゼロ設定されているか、ルータの RSA キーペアが適切に生成されていない可能性があります。また、ユーザが SSHv1 クライアントを使用して接続している SSH サーバが SSHv1 接続を受け入れていない可能性もあります。ホスト名およびドメインを指定していることを確認します。次に、**crypto key generate rsa** コマンドを使用して RSA キーペアを生成し、SSH サーバをイネーブルにします。
- SSHv2 を使用しており、SSH 接続が拒否されている場合は、DSA、RSA ホストキーペアがゼロ設定されている可能性があります。前述の同様の手順に従って必要なホストキーペアを生成し、SSH サーバをイネーブルにしてください。
- ECDSA、RSA または DSA キー ペアを設定する場合、次のエラー メッセージが表示されることがあります。
  - No hostname specified

  hostname コマンドを使用して、ルータのホスト名を設定する必要があります。
  - No domain specified

  domain-name コマンドを使用して、ルータのホスト ドメインを設定する必要があります。
- 使用できる SSH 接続数は、ルータに設定されている仮想端末回線の最大数に制限されます。各 SSH 接続は vty リソースを使用します。
- SSH では、ルータで AAA によって設定されるローカル セキュリティまたはセキュリティ プロトコルが、ユーザ認証に使用されます。AAA を設定する場合、コンソール上で AAA を無効にするためにグローバル コンフィギュレーション モードでキーワードを適用することにより、コンソールが AAA の下で実行されていないことを確認する必要があります。

（注） PuTTY バージョン 0.63 以降を使用して SSH クライアントに接続する場合は、PuTTY 設定の [SSH] > [Bugs] で [Chokes on PuTTYs SSH2 winadj request] オプションを [On] に設定します。これにより、大量の出力が IOS XR から PuTTY クライアントに送信されるたびにセッションが中断する可能性を回避できます。

セキュア シェルの設定

次に、SSHv2 サーバを設定する方法の例を示します。この例では、ホスト名を作成し、ドメイン名を定義し、DSA キー ペアを生成することでルータでのローカルおよびリモート認証に対して SSH サーバをイネーブルにし、SSH サーバを起動し、コンフィギュレーション ファイルを実行するためのコンフィギュレーション コマンドを保存します。

SSH の設定が完了すると、ルータで SFTP 機能が使用できます。

```
configure
hostname router1
domain name cisco.com
exit
crypto key generate rsa/dsa
configure
ssh server
end
```

暗号公開キーと HMAC アルゴリズムを制限する SSH 設定オプション

Cisco IOS XR ソフトウェアには、ルータとの SSH 接続の確立中にピアとネゴシエートされるキーアルゴリズムを制御する新しい設定オプションが用意されています。この機能を使用すると、デフォルトでは無効になっているセキュアでない SSH アルゴリズムを SSH サーバで有効にすることができます。また、新しい設定オプションを使用して、SSH クライアントがルータ上の SSH サーバへの接続中に HMAC アルゴリズムを選択しないように制限することもできます。

暗号のリストをデフォルトの暗号リストとして設定することもできるため、特定の暗号を柔軟に有効または無効にできます。

警告 セキュアでない SSH アルゴリズムを有効にする際には、セキュリティ攻撃の可能性を回避するように注意してください。

HMAC アルゴリズムを無効にするには、XR コンフィギュレーション モードで **ssh server disable hmac hmac-sha1** コマンドを使用します。

必要な暗号を有効にするには、XR コンフィギュレーション モードで **ssh server enable cipher** コマンドを使用します。

次の暗号化アルゴリズムがサポートされています。

- aes128-cbc
- aes256-cbc
- aes128-ctr
- aes256-ctr
- AEAD_AES_128_GCM
- AEAD_AES_256_GCM
- aes128-gcm@openssh.com
- aes256-gcm@openssh.com

暗号の優先順位は次のとおりです。

aes128-ctr、aes192-ctr、aes256-ctr、aes128-gcm@openssh.com、aes256-gcm@openssh.com、aes128-cbc、aes192-cbc、aes256-cbc、3des-cbc。

SSH では、CBC ベースの暗号はデフォルトで無効になっています。これらの暗号を有効にするには、それぞれの CBC オプション（aes-cbc または 3des-cbc）を指定して **ssh server enable cipher** コマンドを使用できます。CTR ベースおよび GCM ベースの暗号はすべてデフォルトで有効になっており、現在、これらの暗号を制御する明示的な設定はありません。

（注） FIPS モードでは、デフォルトで無効になっているアルゴリズムを有効にすることはできません。同様に、これらのアルゴリズムのいずれかを有効にしている場合は、FIPS モードを有効にすることはできません。

HMAC アルゴリズムの無効化

HMAC アルゴリズムを無効にする設定例

```
Router(config)# ssh server disable hmac hmac-sha1
Router(config)#commit
```

実行コンフィギュレーション

```
ssh server disable hmac hmac-sha1
!
```

関連項目

暗号公開キーと HMAC アルゴリズムを制限する SSH 設定オプション （9 ページ）

関連コマンド

- **ssh server disable hmac**

暗号公開キーの有効化

暗号公開キーを有効にする設定例

クライアントとサーバですべての暗号を有効にするには、次のようにします。

ルータ 1：

```
Router(config)# ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc
 aes128-ctr aes128-cbc aes128-gcm@openssh.com aes256-gcm@openssh.com
```

ルータ 2：

```
Router(config)# ssh server algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc
 aes128-ctr aes128-cbc aes128-gcm@openssh.com aes256-gcm@openssh.com
```

クライアントで CTR 暗号、サーバで CBC 暗号を有効にするには、次のようにします。

ルータ 1：

```
Router(config)# ssh client algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

ルータ 2：

```
Router(config)# ssh server algorithms cipher aes128-cbc aes256-cbc aes192-cbc
```

クライアントとサーバで暗号を使用しない場合は、次のようにします。

ルータ 1：

```
Router(config)# no ssh client  algorithms  cipher
```

ルータ 2：

```
Router(config)# no ssh server  algorithms  cipher
```

クライアントとサーバで廃止されたアルゴリズムのみを有効にするには、次のようにします。

ルータ 1：

```
Router(config)# ssh client algorithms cipher aes-cbc 3des-cbc
```

ルータ 2：

```
Router(config)# ssh server algorithms cipher aes-cbc 3des-cbc
```

クライアントとサーバで廃止されたアルゴリズムを有効にし（**enable cipher** コマンドを使用）、CTR 暗号を有効にする（**algorithms cipher** コマンドを使用）には、次のようにします。

ルータ 1：

```
Router(config)# ssh client enable cipher aes-cbc 3des-cbc
Router(config)# ssh client algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

ルータ 2：

```
Router(config)# ssh server enable cipher aes-cbc 3des-cbc
Router(config)# ssh server algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

実行コンフィギュレーション

クライアントとサーバですべての暗号が有効になっている場合：

ルータ 1：

```
ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc aes128-ctr
aes128-cbc aes128-gcm@openssh.com aes256-gcm@openssh.com
!
```

ルータ 2：

```
ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc aes128-ctr
aes128-cbc aes128-gcm@openssh.com aes256-gcm@openssh.com
!
```

関連項目

暗号公開キーと HMAC アルゴリズムを制限する SSH 設定オプション （9 ページ）

関連コマンド

- **ssh client enable cipher**
- **ssh server enable cipher**
- **ssh client algorithms cipher**
- **ssh server algorithms cipher**

セキュア シェルの実装について

SSH を実装するには、次の概念について理解しておく必要があります。

SSH サーバ

SSH サーバの機能によって、SSH クライアントは Cisco ルータに対してセキュアで暗号化された接続を実行できます。この接続は、インバウンド Telnet 接続の機能と同様です。SSH 以前は、セキュリティは Telnet のセキュリティに限定されていました。SSH を Cisco ソフトウェアの認証と併用することで、強力な暗号化が可能になります。Cisco ソフトウェアの SSH サーバは、市販の一般的な SSH クライアントと相互運用できます。

SSH クライアント

SSH クライアント機能は、SSH プロトコルを介して実行されるアプリケーションで、認証と暗号化を行います。SSH クライアントによって、Cisco ルータは他の Cisco ルータ、または SSH サーバを実行する他のデバイスに対して、セキュアで暗号化された接続を実行できます。この接続は、接続が暗号化されている点を除き、アウトバウンド Telnet 接続の機能と同様です。認証と暗号化により、SSH クライアントは、セキュリティ保護されていないネットワーク上でもセキュアな通信を実現できます。

SSH クライアントは、市販の一般的な SSH サーバと連動します。SSH クライアントは、AES、3DES、メッセージダイジェストアルゴリズム 5（MD5）、SHA1、およびパスワード認証の暗号をサポートしています。ユーザ認証はルータへの Telnet セッションで実行されます。SSH がサポートするユーザ認証メカニズムには、Remote Authentication Dial-In User Service（RADIUS）、TACACS+、およびローカルに格納されたユーザ名とパスワードを使用した認証があります。

SSH クライアントでは、発信パケットに DSCP 値を設定することができます。

```
ssh client dscp <value from 0 - 63>
```

設定しない場合は、（クライアントとサーバの両方の）パケットにデフォルトの DSCP 値 16 が設定されます。

SSH クライアントは次のオプションをサポートしています。

- DSCP：SSH クライアント セッションの DSCP 値。

```
RP/0/5/CPU0:router#configure
RP/0/5/CPU0:router(config)#ssh ?
  client   Provide SSH client service
  server   Provide SSH server service
  timeout  Set timeout value for SSH
RP/0/5/CPU0:router(config)#ssh client ?
```

- Knownhost：ローカルデータベースでのホストの pubkey チェックをイネーブルにします。
- Source-interface：SSH クライアント セッションの送信元インターフェイス。

```
RP/0/5/CPU0:router(config)#ssh client source-interface ?
  ATM                 ATM Network Interface(s)
  BVI                 Bridge-Group Virtual Interface
  Bundle-Ether        Aggregated Ethernet interface(s)
  CEM                 Circuit Emulation interface(s)
  GigabitEthernet     GigabitEthernet/IEEE 802.3 interface(s)
  IMA                 ATM Network Interface(s)
  IMtestmain          IM Test Interface
  Loopback            Loopback interface(s)
  MgmtEth             Ethernet/IEEE 802.3 interface(s)
  Multilink           Multilink network interface(s)
  Null                Null interface
```

```
      PFItestmain           PFI Test Interface
      PFItestnothw          PFI Test Not-HW Interface
      PW-Ether              PWHE Ethernet Interface
      PW-IW                 PWHE VC11 IP Interworking Interface
      Serial                Serial network interface(s)
      VASILeft              VASI Left interface(s)
      VASIRight             VASI Right interface(s)
      test-bundle-channel   Aggregated Test Bundle interface(s)
      tunnel-ipsec          IPSec Tunnel interface(s)
      tunnel-mte            MPLS Traffic Engineering P2MP Tunnel interface(s)
      tunnel-te             MPLS Traffic Engineering Tunnel interface(s)
      tunnel-tp             MPLS Transport Protocol Tunnel interface
    RP/0/5/CPU0:router(config)#ssh client source-interface
    RP/0/5/CPU0:router(config)#
```

SSH では、次のようにリモート コマンドを実行することもできます。

```
RP/0/5/CPU0:router#ssh ?
  A.B.C.D  IPv4 (A.B.C.D) address
  WORD     Hostname of the remote node
  X:X::X   IPv6 (A:B:C:D...:D) address
  vrf      vrf table for the route lookup
RP/0/5/CPU0:router#ssh 1.1.1.1 ?
  cipher            Accept cipher type
  command           Specify remote command (non-interactive)
  source-interface  Specify source interface
  username          Accept userid for authentication
  <cr>
RP/0/5/CPU0:router#ssh 12.28.46.6 username admin command "show redundancy sum"
Password:


Wed Jan  9 07:05:27.997 PST
    Active Node    Standby Node
    -----------    ------------
       0/4/CPU0        0/5/CPU0 (Node Ready, NSR: Not Configured)

RP/0/5/CPU0:router#
```

SFTP 機能の概要

SSH には、SSHv2 で導入された新たな標準ファイル転送プロトコルである Standard File Transfer Protocol（SFTP）のサポートが含まれています。この機能は、ルータ設定またはルータ イメージ ファイルをコピーするセキュアで認証された方法を提供します。

SFTP クライアント機能は SSH コンポーネントの一部として提供され、ルータで常にイネーブルになっています。このため、適切なレベルのユーザは、ルータへのファイルのコピーおよびルータからのファイルのコピーが可能です。**copy** コマンドと同様に、**sftp** コマンドは XR EXEC モードでのみ使用できます。

SFTP クライアントは VRF 対応であるため、接続の試行時に特定の送信元インターフェイスに関連付けられた VRF を使用するようにセキュア FTP クライアントを設定することもできます。SFTP クライアントはインタラクティブ モードもサポートしています。このモードでは、ユーザはサーバにログインして特定の作業を UNIX サーバ経由で実行できます。

SFTP サーバは SSH サーバのサブシステムです。つまり、SSH サーバが SFTP サーバ要求を受信すると、SFTP API は SSH サーバに対して子プロセスとして SFTP サーバを作成します。新たな要求のたびに、新しい SFTP サーバ インスタンスが作成されます。

SFTP は、次の手順で新たな SFTP サーバを要求します。

- ユーザが必要な引数を指定して **sftp** コマンドを実行します
- SFTP API は SSH サーバと通信する子プロセスを内部に作成します
- SSH サーバは SFTP サーバ子プロセスを作成します
- SFTP サーバおよびクライアントは暗号化形式で相互に通信します
- SFTP 転送は LPTS ポリサー「SSH-Known」の影響を受けます。ポリサーの値が低いと、SFTP 転送の速度に影響します。

（注） IOS-XR ソフトウェア リリース 4.3.1 以降では、SSH-Known のデフォルトのポリサー値が 2500 pps から 300 pps にリセットされました。この変更により転送速度の低下が予想されます。このパントの原因となる LPTS ポリサー値を高い値に調整することにより、転送速度を上げることができます

SSH サーバが SSH クライアントと新たな接続を確立すると、サーバ デーモンは新たな SSH サーバ子プロセスを作成します。子サーバプロセスは、キー交換とユーザ認証プロセスによって、SSH クライアントとサーバとの間にセキュアな通信チャネルを構築します。SSH サーバがサブシステムを SFTP サーバにする要求を受信した場合、SSH サーバ デーモンは SFTP サーバ子プロセスを作成します。SFTP サーバ サブシステム要求を受信するたびに、新たな SSH サーバ子インスタンスおよび SFTP サーバ インスタンスが作成されます。SFTP サーバはユーザ セッションを認証し、接続を開始します。ユーザのデフォルトディレクトリおよびクライアントの環境を設定します。

初期化が実行されると、SFTP サーバはクライアントからの SSH_FXP_INIT メッセージを待機します。このメッセージは、ファイル通信セッションを開始するためには不可欠です。このメッセージの後に、クライアントの要求に基づいたメッセージが続く場合があります。ここでは、プロトコルは「要求応答」モデルを採用しています。クライアントがサーバに要求を送信すると、サーバはこの要求を処理し応答を送信します。

SFTP サーバは次の応答を表示します。

- ステータス応答
- 処理応答
- データ応答
- 名前応答

（注） サーバは、着信する SFTP 接続を受け付けるために稼働している必要があります。

RSA ベースのホスト認証

サーバの正当性を検証することは、セキュアな SSH 接続を実現する最初の手順です。このプロセスはホスト認証と呼ばれ、クライアントが有効なサーバに接続していることを確認するために実施されます。

ホスト認証はサーバの公開キーを使用して実行されます。サーバは、キー交換フェーズの間に公開キーをクライアントに提供します。クライアントはこのサーバの既知ホストのデータベースと、対応する公開キーをチェックします。クライアントでサーバの IP アドレスが見つからなかった場合は、ユーザに警告メッセージを表示し、ユーザは公開キーを保存するか廃棄するかを選択できます。サーバの IP アドレスは見つかったものの公開キーが一致しない場合、クライアントは接続を終了します。公開キーが有効な場合、サーバは検証され、セキュアなSSH接続が確立されます。

IOS XR SSH サーバおよびクライアントは、DSA ベースのホスト認証をサポートしていました。ただし、IOSなどの他の製品との互換性のため、RSAベースのホスト認証のサポートも追加されました。

RSA ベースのユーザ認証

SSH プロトコルにおいてユーザを認証する方法の 1 つに、RSA 公開キー ベースのユーザ認証があります。秘密キーの保持がユーザ認証の役割を果たします。この方法は、ユーザの秘密キーで作成した署名を送信することで機能します。各ユーザは RSA キーペアをクライアントマシンに保持しています。RSA キーペアの秘密キーはクライアントマシンに残ったままです。

ユーザは、ssh-keygen などの標準的なキー生成メカニズムを使用して、RSA 公開キーと秘密キーのキーペアをUNIXクライアント上に生成します。サポートされているキーの最大の長さは 4096 ビットで、最小の長さは 512 ビットです。次に、一般的なキー生成アクティビティの例を示します。

```
bash-2.05b$ ssh-keygen -b 1024 -t rsa
Generating RSA private key, 1024 bit long modulus
```

公開キーを正常にボックスにインポートするには、公開キーがBase64エンコード（バイナリ）形式である必要があります。インターネットで入手できるサード パーティのツールを使用して、キーをバイナリ形式に変換できます。

公開キーがルータにインポートされると、SSHクライアントは内部で「-o」オプションを使用して要求を指定することで、公開キー認証方式を使用できるようになります。次に例を示します。

```
client$ ssh -o PreferredAuthentications=publickey 1.2.3.4
```

公開キーが RSA 方式によってルータにインポートされていない場合、SSH サーバはパスワードベースの認証を開始します。公開キーがインポートされている場合、サーバは両方の方式の使用を提案します。SSHクライアントはいずれかの方式を使用して、接続を確立します。SSHクライアントからの発信接続の数は 10 まで許可されます。

現時点では、SSH バージョン 2 および SFTP サーバのみが RSA ベースの認証をサポートしています。

（注） 推奨される認証方法は SSH RFC に記載されています。RSA ベース認証のサポートはローカル認証のみです。TACACS/RADIUS サーバに対してはサポートされていません。

認証、許可、およびアカウンティング（AAA）は、Cisco ルータまたはアクセス サーバにアクセス コントロールを設定できる主要なフレームワークを提供する一連のネットワーク セキュリティ サービスです。

SSHv2 クライアント キーボード インタラクティブ認証

キーボードを使用して認証情報を入力する認証方式は、キーボードインタラクティブ認証と呼ばれます。この方式は、SSH プロトコルのインタラクティブな認証方式です。この認証方式では、SSH クライアントは、認証方法の基本的メカニズムを考慮することなく、さまざまな認証方法をサポートできます。

現在、SSHv2 クライアントはキーボード インタラクティブ認証をサポートしています。この認証方式は、インタラクティブなアプリケーションでのみ機能します。

（注） パスワード認証はデフォルトの認証方式です。キーボードインタラクティブ認証方式は、キーボードインタラクティブ認証のみをサポートするようにサーバが設定されている場合に選択されます。