SSH の使いかた for 物理気候のみなさま

はまだあつし@ぶつりきこう

2001/06/24j Physical Climatology Labo. special seminar, revised 2002/06/23

0. 執筆環境

執筆された時点での環境は,

- `stclub01`: Vine Linux 2.5CR / SSH Version OpenSSH_3.1p1
- `stclim01`: Compaq Tru64 UNIX V5.1 / SSH Version 1.2.33
- `stchmd01`: Windows 2000 Professional SP2 / SSH Secure Shell Version 3.1.1 / ASTEC-X Version 3.11

となっています. 本文書は物理気候計算機群に特化した環境を前提としており, SSH サーバ・クライアント共に設定次第でどうにでも変わることに留意してください.

例として挙げてあるのは全て

「hamada さんが stclub01 に居て, stclim01 へ SSH1 を使って云々する」

という状態を想定しています.

なお, この文書に登場する人物, ホスト等は実在する人物等と一切関係あります（笑）

1. SSH とは？

SSH (Secure SHell) は, ネットワーク上の 2 つのホスト間に安全な通信経路を提供します. SSH コマンド群は, `rlogin`, `rsh`, `rcp` といった所謂'r' コマンドの置き換えになっており, 各々に対応する's' コマンドは `slogin(ssh)`, `ssh`, `scp` です. 's' コマンドが'r' コマンドに対し有利な点は,

1) 通信が暗号化されている
   ・これは言うまでもありません.

2) .rhosts による認証を用いない
   ・所謂 'IP スプーフィング（spoofing; なりすまし）' を防ぐことができます.
   ・`xhost` コマンドも必要ありません. 安全.

3) 接続先でも `DISPLAY` 環境変数を保存する
   ・telnet や rlogin した後のように, '`setenv DISPLAY` 〜' などとする必要はありません. 楽.

1.1 注意すべき点

SSH は telnet や r 系コマンドに比べて安全ですが, 以下の点には注意しておく必要があります.

- passphrase の選び方に注意！
  ・SSH の主たる安全性は「ログインパスワードよりも使用可能文字数が長いから解読しにくい」という点にありますので, passphrase を選ぶときの注意はログインパスワードの時と同じです. 簡単な英文ですとか, 盗まれやすいものは避けましょう.
- slogin した後でも telnet しちゃダメ！
  ・SSH と関係ない（SSH が監視していない）通信については, これまで通り平文が流れていることに注意しなければなりません[1].

[1] 実は ftp, POP3, SMTP などもこれに抵触しています. これらはポートフォワードなどによって回避可能ですが, まだ*初心者向けの*

2. 取り敢えず使ってみる〜ログインパスワード認証〜

何はともあれ, まずは使ってみましょう.

2.1 SSH で login

お使いのターミナルで,

兎に角 login

```
hamada@stclub01% ssh stclim01
```

と打ち込みます[2]. 相手側と初めて接続する場合は,

あなたホントにいいの？

```
The authenticity of host 'stclim01 (10.xxx.xxx.xxx)' can't be established.
RSA1 key fingerprint is xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx.
Are you sure you want to continue connecting (yes/no)?
```

などと聞いてきますが, これは「本当に接続しますか？」という確認ですので, yes と答えます. すると,

パスワード入力

```
Warning: Permanently added 'stclim01,10.xxx.xxx.xxx' (RSA1) to the list of known hosts.
hamada@stclim01's password:
```

と, 接続先のログインパスワードを聞いてきます. パスワードを正しく入力すれば, 接続が確立されます. 一見したところ telnet や rlogin と変わりませんが, 今送信したパスワードも含めた全ての通信が暗号化されています.

2.2 リモートホストのコマンド実行

全て'r' コマンドの置き換えだと思えばいいんですね. ですから例えば, stclim01 で走っている hamada さんのプロセスを見たいなーと思えば,

ssh の例

```
hamada@stclub01% ssh stclim01 ps auxw | grep hamada
hamada@stclim01's password: ←パスワードを入力
hamada   15612  0.0  0.1 2.33M 344K ?? S  21:30:44   0:00.04 tcsh -c ps auxw
```

としてやればいい訳です.

ssh コマンドは, xon コマンドの変わりも務めます. 例えば stclub01 の XEmacs を使うなら,

X プログラムとの組み合わせ

```
hamada@stclim01% ssh -f stclub01 xemacs
hamada@stclub01's password: ←パスワードを入力
```

ここではオプション'-f' は X プログラムを使う際のおまじないだと思って下さい.

2.3 scp

これも rcp と同じように使います. 転送速度なんかの情報が出て, ちょっとかっこいいです. 例↓

解が整理できてないので, 今回は触れないことにします.

[2]実は slogin は次に述べる ssh へのシンボリックリンクになっています. ですので, その日の気分で使い分ければよいでしょう.

scp の例

```
hamada@stclub01% scp stclim01:~/.cshrc ./.cshrc_stclim01
hamada@stclim01's password:
.cshrc               100% |*****************************|  3031       00:00
```

3. RSA 鍵認証

ログインパスワードを使った認証でもいいのですが, より強力な認証方法を使うことも出来ます. SSH では,「公開鍵暗号」というものを用いて認証が行なわれます. その際,「秘密鍵」と「公開鍵」のペアが作成され, 名前の通り秘密鍵はパスフレーズを設定して死守し, 公開鍵は別に見られてもいい, という関係になっています.「鍵」と「鍵穴」みたいなもんです. 鍵穴を見られたって, *鍵を作らない限り*家に侵入されることは無い訳ですから.

3.1 鍵の作成 ([1]p.144)

パスフレーズが盗まれてしまっては元も子もありませんから, 鍵の作成作業はセキュアな環境で行ないましょう. 今回挙げる例ですと, hamada さんが

- stclub01 の前に座っているとき — 後ろから盗み見されていなければ大丈夫.
- そうでないとき — slogin で stclub01 に入ってから作業しましょう.

RSA 鍵を作るためには, `ssh-keygen` コマンドを実行します.

ssh-keygen の例

```
hamada@stclub01% ssh-keygen -t rsa1
Generating public/private rsa1 key pair.
Enter file in which to save the key (/usr/users/hamada/.ssh/identity): ←何も入力せず
Enter
Created directory '/usr/users/hamada/.ssh'.
Enter passphrase (empty for no passphrase): ←秘密鍵を開くためのパスフレーズ入力
Enter same passphrase again: ←もいっかい
```

パスフレーズ[3]には文字数制限はありませんから, *忘れない程度に*他人に分からないものを選びましょう ([1]p.148). '`-t rsa1`' というオプションは, SSH1 用の RSA 鍵を作ることを意味します. stclim マシンで鍵を作成するときは, このオプションは必要ありません.

パスフレーズの入力に成功すれば, 以下の様なメッセージと共に, 鍵が作成されます.

作成成功

```
Your identification has been saved in /usr/users/hamada/.ssh/identity.
Your public key has been saved in /usr/users/hamada/.ssh/identity.pub.
The key fingerprint is:
xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx hamada@stclub01
```

ここで作られた `identity` が秘密鍵, `identity.pub` が公開鍵です.

パスフレーズを変更するときは, `-p` オプションを付けて実行します. ログインパスワードの変更と要領は同じです.

[3] pass'word' だと何だか 1 単語だけしか使えない様な気がするので, pass'phrase' という言葉が使われています.

passphrase 変更

```
hamada@stclub01% ssh-keygen -t rsa1 -p
Enter file in which the key is (/usr/users/hamada/.ssh/identity): ←何も入力せず Enter
Enter old passphrase: ←変更前の passphrase
Key has comment 'hamada@stclub01'
Enter new passphrase (empty for no passphrase): ←変更後の passphrase
Enter same passphrase again: ←もいっかい
Your identification has been saved with the new passphrase.
```

3.2 鍵の公開〜RSA 認証を行なうための設定〜

鍵を作っただけでは, この鍵を使って他のマシンに SSH でログインすることは出来ません. ログイン先のマシンに公開鍵を教えてやり,「鍵穴」を作る必要があります. それには, 接続元の公開鍵を接続先の`~/.ssh/authorized_keys`に追加してやればよいです.

まず, 接続元の公開鍵を接続先にコピーします.

公開鍵をコピー

```
hamada@stclub01% scp .ssh/identity.pub stclim01:~/
hamada@stclim01's password: ←パスワードを入力
identity.pub          100% |*****************************|   353       00:00
```

接続先にログインし, 先ほどコピーしてきた`identity.pub`を`~/.ssh/authorized_keys`に登録します. `~/.ssh`が無いときは作ります. 既に`~/.ssh/authorized_keys`がある場合, 後ろに追加していけばいい訳です.

公開鍵の登録

```
hamada@stclub01% ssh stclim01
hamada@stclim01's password: ←パスワードを入力
hamada@stclim01% cat identity.pub >>! .ssh/authorized_keys ←名前間違えないでね
hamada@stclim01% rm identity.pub ←後始末
```

これで鍵の登録が終わりました. 早速ログインしてみましょう.

RSA 認証でログイン

```
hamada@stclub01% exit
hamada@stclub01% ssh stclim01
Enter passphrase for RSA key '/usr/users/hamada/.ssh/identity': ←ここがさっきと違う
```

パスフレーズを正しく入力すれば, 接続が確立します.

stclub 同士で RSA 認証を使ってログインしたいときは, '`-1`（数字の 1）' オプションをつけます.

3.3 ssh-agent

いちいちパスワードを打つのが面倒な人は, `ssh-agent` を利用するとよいでしょう.

agent の起動

```
hamada@stclub01% exec ssh-agent $SHELL
```

新しくシェルが起動します. そうしておいて, 次に秘密鍵をメモリ上に登録します. これには `ssh-add` を使います.

鍵の登録

```
hamada@stclub01% ssh-add
/usr/users/hamada/.ssh/id_rsa: No such file or directory ←気にしないで OK
/usr/users/hamada/.ssh/id_dsa: No such file or directory ←気にしないで OK
Enter passphrase for hamada@stclub01.kugi.kyoto-u.ac.jp: ←パスフレーズを入力
Identity added: /usr/users/hamada/.ssh/identity (hamada@stclub01)
```

一度秘密鍵が登録されたら, このコマンドを実行したシェル上で, 以後 ssh, scp で一々パスフレーズを要求されなくなります.

3.4 X のセッションを丸々ssh-agent に登録

3.3 で述べた方法では, 新しく kterm や emacs を起動する度に ssh-agent → ssh-add しなければならず, 面倒です. X のセッション全体を ssh-agent 配下に置くことで, この問題を回避できます. それには, これまで使ってきた~/.xsession を~/.xinit に名前を変えて, 新しく次のような~/.xsession ファイルを用意します ([1]p.162).

~/.xsession の例

```
#!/bin/sh
SSH_AGENT="ssh-agent"
SSH_ADD="ssh-add < /dev/null"

if [ -x $HOME/.xinit ]
  then exec $SSH_AGENT /bin/sh -c "$SSH_ADD; exec $HOME/.xinit"
else xterm
fi
```

~/.xinit と~/.xsession のモードを 755 にしておくことを忘れずに. こうしておけば, xdm ログインの後, パスフレーズを入力する画面になります. パスフレーズが正しく入力されれば, 以後パスフレーズの入力は省略されます.

4. Windows, Macintosh で SSH を使う

これまでは UNIX 計算機間での話でしたが, Windows や Mac に ASTEC-X のような X サーバを載せて使ってる人も居ますよね. その場合セキュアな X の環境を確保するのには, ちょっと工夫が要ります. ここでは大まかな手順を述べるに留めます.

まず, X サーバを起動します. XDMCP は使わない設定にしておいて下さい. Windows や Mac の SSH クライアントから, X 環境を起動したいマシンに slogin します. X11 forwarding を有効にしておいてください[4]. 次に, DISPLAY 環境変数を設定し[5], 3.4 で作成した~/.xsession を実行します. あとは 3.4 節と同じ手順です.

筆者の PC は DHCP 環境下にあり, いちいち IP アドレスを調べるのが面倒なので, こんなスクリプトを使っています.

[4]例えば Windows の SSH Secure Shell クライアントなら メニューバーから [Edit] → [Settings] と選んで、設定ウインドウの [Tunneling] で [Tunnel X11 connections] にチェックを入れる.

[5]ほんとは要らない筈なんですが、壁紙など一部で不具合を確認した為、念のため設定しておきます.

例えばこんな感じ

```
#!/bin/bash
export DISPLAY=‘echo $SSH_CLIENT | awk ’{print $1}’‘:0.0
exec $HOME/.xsession &
```

これを~/startSessionという名前で保存し, UNIX ホストに SSH ログインしたあと最初に実行しています.

5. SSH1 に関係するファイル (ユーザ編)

SSH を使っていると~/.sshディレクトリにはいくつかのファイルが作成されます. ここではそれらのファイルの意味を簡単に解説します.

- authorized_keys
  このホストに RSA 認証を用いてログインしたい, ログイン元ホストの RSA 公開鍵を登録します.
- identity
  ユーザの RSA 秘密鍵です. バイナリファイルになっていて, ユーザ本人だけが読み書き出来ます. 一度作った後は触ることはありません. 絶対に他人に知られてはいけません.
- identity.pub
  ユーザの RSA 公開鍵です. テキストファイルです. ログインしたいホストの~/.ssh/authorized_keysファイルに追加します.
- known_hosts
  1 度ログインしたホストのホスト公開鍵が登録されます. このファイルに登録された鍵が相手ホストのホスト鍵と異なる場合, 以下のような警告が出てログインできません.

  盗聴!?

  ```
  hamada@stclub01% ssh stclim01
  @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
  @    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
  @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
  IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
  Someone could be eavesdropping on you right now (man-in-the-middle attack)!
  It is also possible that the RSA1 host key has just been changed.
  The fingerprint for the RSA1 key sent by the remote host is
  xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx.
  Please contact your system administrator.
  Add correct host key in /usr/users/hamada/.ssh/known_hosts to get rid of this message.
  Offending key in /usr/users/hamada/.ssh/known_hosts:1
  RSA1 host key for stclim01 has changed and you have requested strict checking.
  Host key verification failed.
  ```

  SSH の再インストールなどによってもホスト鍵は変わります. もしこのメッセージが表示されたときは, 管理者に鍵を新しくしたか確認しましょう. 確認できたら, known_hosts ファイルをエディタで開き, 該当するホストのホスト公開鍵を削除して再接続してください.
  管理者が鍵を新しくしたのでなければ, 悪意を持った何者かがあなたの通信を盗聴している可能性があります (man-in-middle attack).
- random_seed
  乱数生成の種です. ユーザが触ることはありません.（OpenSSH では作成されないようです.）

以上のファイル群は, 基本的に他人に見せる必要のないファイルですから, ユーザのみにアクセス権を与えるようにしておくのが望ましいです[6].

ファイルモード

```
hamada@stclub01% ls -la ~/.ssh
drwx------    2 hamada       4096 Jun 23 05:07 .                ←ディレクトリは 700
drwxr-xr-x   20 hamada       4096 Jun 23 05:34 ..
-rw-------    1 hamada       1355 Jun 21 02:33 authorized_keys  ←ファイルは 600
-rw-------    1 hamada        213 May 30 15:26 authorized_keys2
-rw-------    1 hamada        549 Jun 10 09:34 identity
-rw-------    1 hamada        353 Jun 10 09:34 identity.pub
-rw-------    1 hamada       2842 Jun 23 05:29 known_hosts
```

References

A. Carasik 著, まえだひさこ監修, 2000: SSH セキュアシェルリファレンス, 翔泳社, 243pp.

KUINS ニュース 33, 2000, http://www.kuins.kyoto-u.ac.jp/news/33/ssh.html.

http://www.google.com/ で 'SSH 使い方' ってな感じで検索（約 4000 件）.

man ssh とか :-)

[6]（私的意見）identity.pub は他人に知られても構いませんが, 誰にでも見せてしまうのは気持悪くないですか？