SSH : Secure Shell

~ おでかけ前に鍵かけて ~

伊東栄典*

九州大学大型計算機センター (以下センター) のスーパーコンピュータ VPP700 (kyu-vpp) ,汎用計算機 (kyu-cc) ライブラリサーバ (wisdom) に SSH(**S**ecure **Sh**ell) をインストールしました . 1999 年 3 月 31 日現在 , SSH には非商用バージョンの SSH1 と , 改訂された SSH2 があります . センターの計算機にインストールしているのは SSH1 (Ver.1.2.26) です .

本稿では SSH の仕組の簡単な説明と , センター計算機での SSH 利用方法について説明します . なお , SSH の詳細な説明は公式 WWW ページ (http://www.ssh.fi/) を参照して下さい . また , 以下の説明では特に指定の無い場合 , SSH は SSH1 の事であるものとします .

1 はじめに

1.1 通信路の安全性

センター利用者の皆様は , インターネットにおける通信内容は他者に傍受される可能性がある事を御存知でしょうか . 例えば通信にイーサネットを用いる場合 , 他の計算機から通信内容を傍受することが可能です . イーサネットは同じ通信線を共有して通信を行なう CSMA/CD (Carrier Sense Multiple Access with Collision Detection) という方式を用いています [1] . CSMA/CD における受信は , 常にネットワーク上の通信を見張っていて自分宛の通信であれば内容を取得するという方式で行ないます . 送信では , 誰も送信をしていない場合は直ちに信号送信 , 既に通信を行なっている計算機が存在する場合 , ある時間間隔 (ランダムに決定) 待機後に再度送信 , という方法で行ないます . このような方式であるため , 同一イーサネット領域に接続している計算機であれば , 他の計算機への通信を傍受することが可能です .

またイーサネットでなくても計算機やネットワークに詳しい人であれば通信内容を傍受できる可能性があります . インターネットでは , 通信内容は様々な組織を経由してバケツリレー的に運ばれています . 通信の途中経路に存在する全ての組織が信頼できるわけではありません . 通信の途中経路に悪意のあるネットワーク管理者が存在すれば通信データを覗くことができます .

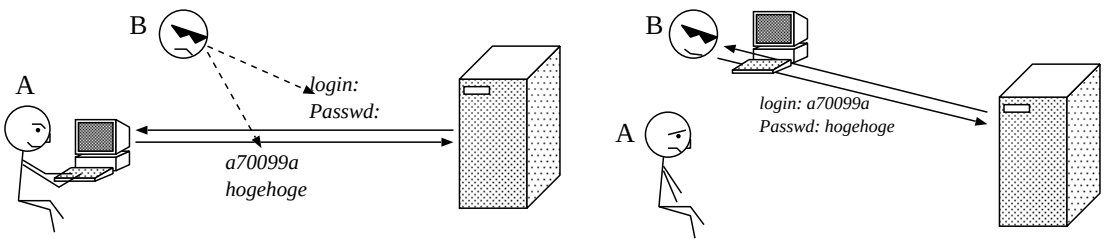

図 1: 盗聴の危険性

現在 , 遠隔地から telnet 接続でセンターの計算機を利用する場合 , 利用者の認証は , 利用者名 (利用課題番号) とパスワードの対を用います . 通常の telnet では通信内容は平文 (暗号化されていない文字列) として送受信されます . このため , 通信経路のネットワークを監視している計算機があれば , 通信内容を傍受して利用者名とパスワードの対を取得する事が可能です . 例えばセンター利用者の A さんの利用者名 (利用課題番号) とパスワードを , 悪意を持つ B さんが通信内容を傍受して取得した場合 , B さんは A さんになりすましてセンターの計算機を利用する事が可能です . この場合 , B さんが計算機を利用したのに , A さんへ利用負担金が課せられてしまいます .

*九州大学大型計算機センター
E-mail:itou@cc.kyushu-u.ac.jp, http://www.cc.kyushu-u.ac.jp/RD/itou/

利用負担金の問題のほかにも，B さんが A さん名義で世界中の人に悪意のあるイタズラメールをばらまいたり，不正なメール中継に利用するかもしれません．A さんになりすまして，ログイン先のシステムを改変，破壊という可能性もあります．また他人になりすましてログインし，そこを足掛かりにして，他の計算機を攻撃する可能性もあります．

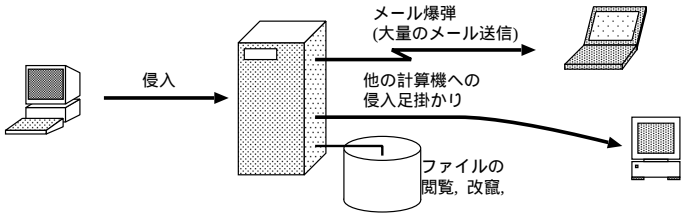

図 2: 様々な攻撃

1.2 暗号方式の概要

このような行為を防ぐために，通信内容の暗号化や，接続する計算機または利用者の認証方式について研究と実用化が進んでいます．今回解説する SSH は公開鍵暗号方式と共通鍵暗号方式を用いて，認証と通信内容の暗号化とをするもので，UNIX における遠隔手続呼出である r コマンド群 (`rcp`，`rlogin`，`rsh`) の上位互換プログラム群です．SSH では認証方式として，RSA 公開鍵暗号方式を用いています [6, 7, 8]．既に様々な論文や本 [4, 5] で RSA 暗号方式の説明が行なわれていますので，ここでは簡単な概略のみを説明します．

通信内容の文を s とすると，この s をそのまま送ると平文による通信になります．これに対して暗号通信では，送信元で s をある関数 f により変換 (暗号化) して得られる暗号文 $c = f(s)$ を送信します．受信元では受けとった c を復号化関数 $g(c)$ で解読して，通信内容 s を取得します．これらの暗号方式は，大きく共通鍵暗号方式と公開鍵暗号方式とにわけられます．

共通鍵暗号方式

まずは共通鍵暗号方式について説明します．共通鍵暗号方式とは，図 3 に示しているような暗号化と復号化の際に同じ鍵を用いる暗号方式で，古くから使われてきた暗号方式です．簡単な共有暗号鍵方式には，シーザー暗号方式があります．これは，アルファベットを k 個だけシフトするといったものです．暗号化関数 $f(s, k)$ は文字列 s を k シフトする関数と言えます．復号化関数は k だけ逆にシフトする関数になります．例えば，“`I love You.`” という文を送りたい場合に，1 つずらして “`J mpwf Zpv.`” とします．この場合は 1 つシフトしているので $k = 1$ になります．上記のシーザー暗号のように，共通鍵暗号方式では送信者と受信者の間で同じ鍵を共有して暗号化および復号化を行ないます．

図 3 のように，Bob が Alice に文を送ることを例にします．まず，最初に両者の間で用いる暗号方式 (暗号化/復号化関数) と，共通して用いる暗号鍵を決定します．Bob が Alice に文章 s (“今日遊びに行こう”) を送る場合，二人の会話を他人に盗聴されないように暗号化するものとします．Bob は先に取り決めた暗号化関数 f と暗号鍵 k を使って暗号文 $c = f(s, k)$ を作成し，この暗号文を Alice に送信します．Alice は受けとった暗号文を，すでに保持している共通暗号鍵 k で復号化します．暗号化関数，復号化関数，共通鍵についての合意がとれているので，問題なく復元できます．

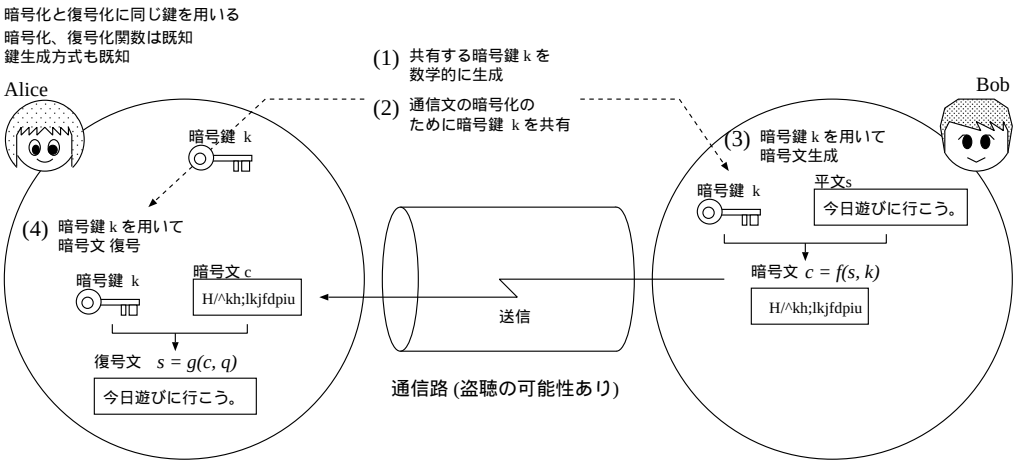

図 3: 共有鍵暗号方式

共通鍵暗号方式の問題は，その鍵の共有方式にあります．インターネットのように盗聴の危険性のある通信路では，通信により暗号鍵を送ると，鍵を盗聴される可能性があります．共通鍵暗号方式では，暗号化と復号化に同じ鍵を用いるため，鍵がバレてしまうと暗号化による安全性は根本から壊れてしまいます．

公開鍵暗号方式

次に公開鍵暗号方式について説明します．公開鍵暗号方式では，暗号化と復号化に用いる鍵が異なります．Alice への通信を行なう場合，Alice は暗号に用いる鍵を二つ用意します．片方の鍵を通信相手に渡す公開鍵 (public key) とし，もう一方の鍵を Alice だけが知る秘密鍵 (private key) とします．ここでは p を公開鍵，q を秘密鍵とします．

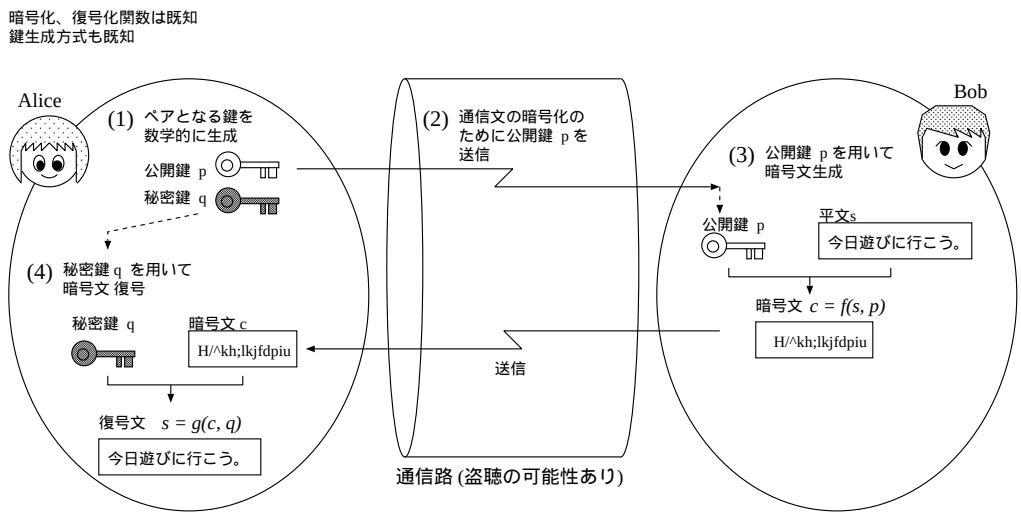

図 4: 公開鍵暗号方式

Bob が Alice へ平文 s を暗号文 c にして通信する場合，暗号化関数 f と Alice の公開鍵 p を用いて暗号文 $c = f(s, p)$ を作成します．この c を Alice へ送ります．受信者である Alice は，復号化関数を用いて暗号文 c を復号化するわけですが，その際には自分だけが知っている秘密鍵 q を用いて復号化します．復号化関数が g であるとすると，元の通信内容である平文 s は $s = g(c, q)$ という関数で与えられます．この際，暗号文 $f(s, p)$ は秘密鍵 q でなければ復号化できない所に公開鍵暗号方式の強さがあります．

暗号化関数 f と復号化関数 g により，様々な種類の公開鍵暗号方式が提案されています．また同じ暗号方式であっても，鍵 k の長さにより暗号の強度 (解読されにくさ) が変化します[1]．ただし，どちらの暗号方式でも暗号の強度が鍵の秘匿性にかかっているのは同じです．共通暗号鍵方式では，鍵データを盗聴の可能性のある通信路上にそのまま流すわけにはいきません．鍵を盗聴されれば，暗号化と復号化に同じ鍵を用いるため他者にも暗号文の解読が可能になってしまうからです．しかし，暗号化および復号化にかかる手間が少ないため処理時間が早いという利点があります．これに対して公開鍵暗号方式では，秘密鍵は同様に秘匿しなければなりませんが，公開鍵は他者に知られても問題ありません．通信路上に公開鍵データを流しても良い分だけ，公開鍵暗号方式がインターネットに向いているといえます．しかし現在の公開鍵暗号方式は暗号化に要する処理時間が長いという問題があります．

2 SSHの仕組

前節で述べたように，暗号には共有鍵方式と公開鍵方式があります．SSH はそれぞれの暗号方式の特性を活かして両方式を利用しています．この章では 2 つの計算機間での SSH を用いた暗号化通信について述べます．

2.1 利用する鍵

まず，認証と暗号化通信を行なうための鍵について説明します．鍵とは実際には数値です．SSH では図 5 に示している鍵を用います．SSH でも通信を行なう 2 つの計算機をクライアントとサーバに分けて考えるモデルを使っています．通信を受付ける計算機がサーバとなり，通信を行なうための接続要求を出す計算機がクライアントになります．サーバ側の鍵とクライアント側それぞれに，計算機固有の鍵 (ホスト鍵)，利用者固有の鍵 (ユーザ鍵) があります．また，鍵を登録していない場合には，利用者認証するためにパスワードを用いる場合もあります．

必要な鍵	Hp , Hq	サーバとなる計算機固有のホスト鍵
	Sp , Sq	一時間毎に生成されるサーバ鍵
	K	セッション鍵
無くても良い鍵	Cp , Cq	クライアントとなる計算機固有の**RSA**鍵
	Up , Uq	利用者の**RSA**鍵

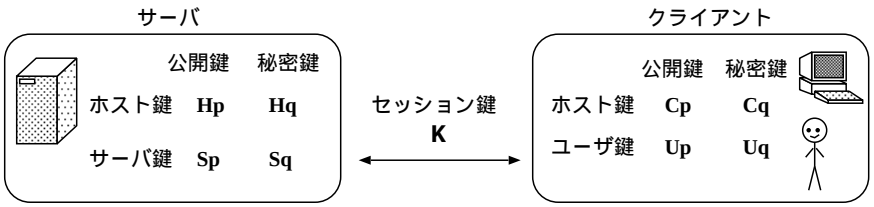

図 5: 利用する鍵

2.2 認証

SSH では通信開始時に，通信相手が正当な実体 (人，プログラム，計算機) であるのかどうか，という認証を行ないます．人あるいは計算機が正しく認証されたならば，暗号化した通信を開始します．SSH では，次の 3 つの実体が正しいものであるかどうかを認証します．

- ホスト認証：接続する計算機が正当な相手であるのかどうかの認証
- ユーザ認証：接続しようとしている利用者が，正当な利用者であるかどうかの認証
- 公開鍵の検証：公開鍵が有効であるかどうかの検証

[1]全ての鍵候補で復号化を試せば解読可能です．そのため候補の多い長い鍵を用いた方が安全です．16bit 長の鍵なら候補数は約 6 万 5 千通りですが，32bit なら約 42 億，128bit なら... ですから．

SSH では，図 6 に示した RSA 公開鍵暗号方式を用いた認証方式を使います．この認証では，RSA 公開鍵暗号方式の暗号化関数と復号化関数が同一関数であるという対称性を利用しています．この関数を RSA，公開鍵を p，秘密鍵を q とすると，平文 s を公開鍵 p で暗号化してできる暗号文 c は，$c = RSA(s, p)$ になります．暗号文 c の復号化は，同じ RSA 関数と公開鍵に対応する秘密鍵 q とを用いて $RSA(c, q)$ で元の平文 s を得ます．逆に秘密鍵 q で暗号化した平文は，対応する公開鍵で復号化できます．この性質を利用すると，文字列を送り合うことにより正しい相手であるかどうかを認証する事が可能です．

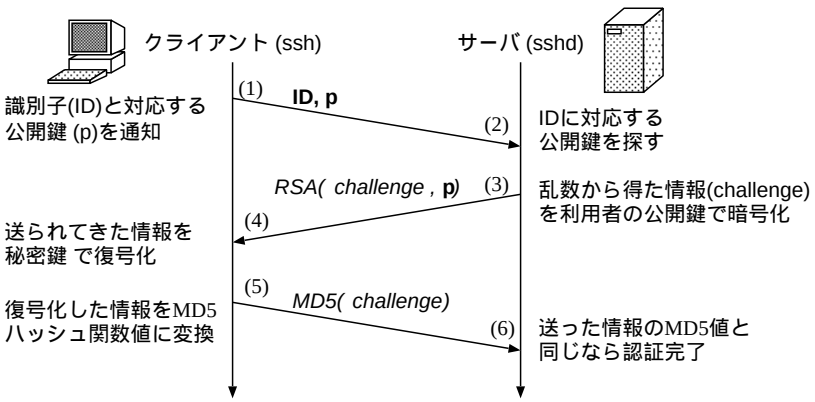

図 6: RSA 公開鍵暗号方式を用いた認証

図 6 で示した認証方式を利用して，SSH では以下に示す認証方式を提供しています．

(a) RSA 認証

最初にサーバとクライアントホスト鍵を用いて図 6 の方式で，計算機の認証を行ないます (ホスト RSA 認証)．次に，利用者の鍵を利用して利用者の RSA 認証を行ないます．実際には図 7 に示すような手順で認証作業が行なわれます．

(b) ホスト RSA 認証付き rhosts 認証

最初にサーバクライアント間でホスト RSA 認証を行ないます．その後，rsh と同様に，サーバ側 (ログイン先) の ~/.rhosts，~/.shosts，/etc/hosts.equiv，/etc/shosts.equiv のいずれかにクライアント (ログイン元) の計算機名が記述されており，かつサーバ側の/etc/ssh_known_host，~/.ssh/known_hosts のいずれかにクライアント側のホスト鍵が記述されていれば，正当な利用者であると認証し，ログインが許可されます．

(c) パスワード認証

パスワード認証は (a)RSA 認証を利用しておらず，(b) ホスト RSA 認証付き rhosts 認証の条件を満していない場合におこなわれます．最初にサーバクライアント間でホスト RSA 認証を行ないます．ここで通信路の暗号化が始まります．次に，telnet と同様にクライアント側の利用者はユーザ名とパスワードを入力します．サーバはユーザ名とパスワードの対を用いて認証を行ないます．通信は暗号化されますので，telnet のように平文でユーザ名やパスワードが通信路を流れることはありません．

(d) rhosts 認証

rhosts 認証は既存の rsh や rlogin のようにサーバ側の ~/.rhosts ファイルなどに記載されている計算機からのアクセスは無条件に許可する方式です．普通に SSH をインストールした場合，rhosts 認証による接続はできないようになっています．センターの計算機でも rhosts 認証は許可していません．

2.3 通信確立手順

図 7 に SSH のセッション確立手順を示します．図に示している以外に途中でいくつかの分岐があります．例えばサーバとなる計算機が SSH を使用できない場合には，自動的に他の手法を選択します．ssh ならば rsh，scp の場合は rcp, slogin の場合は rlogin を，という風に SSH コマンドの代りに r 系コマンドになります．

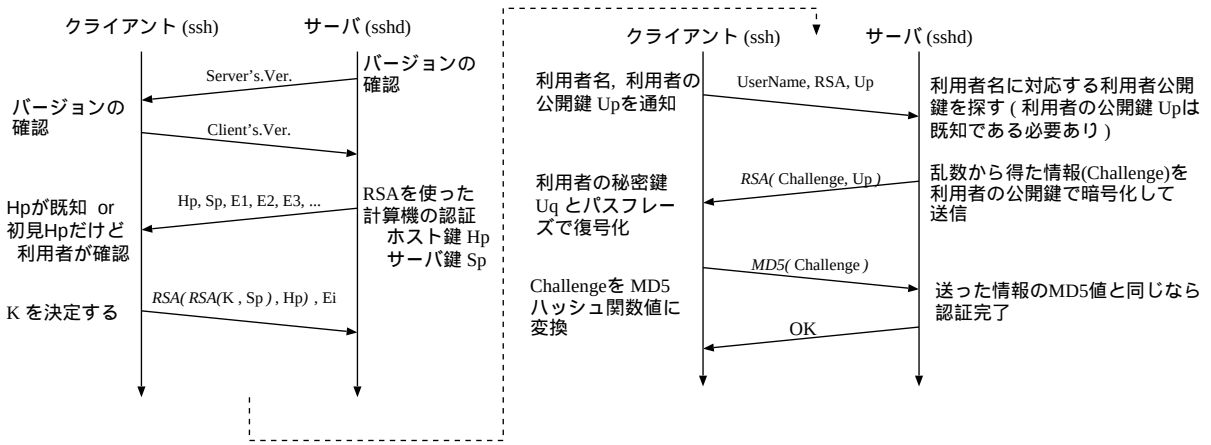

図 7: セッション確立までの手順

サーバとクライアントとの間で，バージョン番号の確認および計算機の認証 (ホスト認証) が成功した場合，その通信セッションで用いる共通鍵を動的に決定します．決定した共通鍵の交換は，公開暗号鍵方式を用いて両者の間で交換されます．以後，そのセッション内の通信は交換した鍵を使った共通鍵暗号方式で通信内容を暗号化します．通常，IDEA，Blowfish，トリプル DES，といった暗号方式を用います．また ARCFOUR(RC4-128 互換)，TSS といった暗号方式にも対応しているようです．

共通鍵の交換は RSA 公開鍵暗号方式を用いて行なわれます．通信で使用された共通鍵データは毎回破棄されます．セッションの通信を暗号化することにより，計算機の IP 偽装 (なりすまし) を防御できます．計算機の RSA 認証により，DNS 偽装やルーティング偽装を防御することができます．

3 SSH の利用 (UNIX)

二つの計算機間の通信に SSH を利用するためには，通信を行なう計算機の両方に SSH をインストールする必要があります．この節では UNIX 系 OS における SSH の利用方法について説明します．Windows95/98 でも SSH を利用する事は可能です．Windows95/98 での SSH の利用については第 4 節を参照して下さい．

3.1 ソースコードの入手

SSH のソースコードはインターネットを通じて様々な所から入手可能です．以下に SSH のソースコードを配布している ftp サイトを示します．

- 一次配布元：`ftp://ftp.cs.hut.fi/pub/ssh/` [2]
- 国内：`ftp://ftp.kyoto.wide.ad.jp/pub/security/ssh/`

古いバージョンはセキュリティに問題があるので，常に最新バージョンを入手するようにして下さい．1999 年 3 月 31 日現在の最新バージョンは `ssh-1.2.26.tar.gz` です．

SSH のソースコードは上記の ftp サイト以外からも入手できますが，国外からの入手および国外での利用には注意が必要です．SSH の `README` ファイルの記述によりますと，フランス，ロシア，イラク，パキスタンでは，特

[2] フィンランド，Helsinki University of Technology.

に許可がない限り暗号プログラムの利用は違法になります．また米国へ SSH を持ち込む事はできますが，犯罪防止のために米国内からの暗号プログラム輸出は禁止されているようです．SSH はフィンランドで開発されましたので日本での利用に問題はありませんが，米国サイトからのソースコード入手をしないように注意して下さい．

3.2 インストール

研究室あるいは個人の計算機と，センター計算機との間で SSH を利用するためには，SSH クライアントだけ用意すれば十分です．自分の UNIX サーバを SSH を用いて遠隔地から利用する場合には，SSH サーバをインストールする必要があります．インストール作業について説明します．

最初に SSH のファイル一式である `ssh-1.2.26.tar.gz` を，gzip や tar コマンドを用いて展開します．すると `ssh-1.2.26/` ディレクトリができ，その中にファイル群が展開されています．その中の `INSTALL` ファイルにインストール方法が説明されていますので，その記述に従ってインストール作業を行ないます．インストールは `configure` スクリプトに基づくものです．有名な UNIX 系 OS の場合，ルートになって次の 3 つのコマンドを入力すれば，インストールできるでしょう．インストールの詳細については，ソースに附属する文書を参照して下さい．

```
# ./configure
# make
# make install
```

上記の作業途中に，プログラムのインストール先などについて質問されます．それらの質問に対して特定の指定なしに作業を進めた場合，次のコマンド群がディレクトリにインストールされます．sshd は`/usr/local/sbin/` に，それ以外は`/usr/local/bin/`ディレクトリにインストールされます．

ssh, slogin, scp,	SSH クライアントプログラム
sshd,	SSH サーバプログラム
ssh-keygen, ssh-agent, ssh-add, ssh-askpass, make-ssh-known-hosts	その他ユーティリティ

インストール時に設定ファイルおよび RSA ホスト鍵も生成されます．

`/etc/ssh_config`	SSH の設定ファイル
`/etc/sshd_config`	sshd の設定ファイル
`/etc/ssh_host_key`	ホスト鍵（秘密鍵）
`/etc/ssh_host_key_pub`	ホスト鍵（公開鍵）

以上でインストール作業は終了です．

3.3 SSH サーバ (sshd) の起動

出張先などの遠隔地から，自分の研究室の計算機を SSH で利用するような場合には，計算機を SSH サーバとして稼働しておかなければなりません．そのためには sshd プログラムをデーモンとして起動しておきます．試しに sshd を実行するだけでしたら，ルート権限のあるユーザになり，sshd を直接実行すれば良いでしょう．常に sshd を使う場合には，計算機起動時に自動的に sshd も起動されるようにデーモン起動スクリプトを用意しておいた方が良いでしょう．参考として，私が用いているスクリプトを記述しておきます．Sun Solaris2.5.1 上で利用しており，`/etc/rc2.d/`ディレクトリに実行権限を付けたファイル (ファイル名 `S99sshd` [3]) として置いています．

[3] Solaris2.5.1 では/etc/rc2.d/下に大文字 K や S で始まるスクリプトファイルを置くと，起動時に実行されるようになっています．

```
#!/bin/sh

case "$1" in
'start')
        if [ -f /usr/local/sbin/sshd ]; then
                echo "Starting SSH server."
                /usr/local/sbin/sshd
        fi
        ;;
'stop')
        if [ -f /etc/sshd.pid ]; then
                echo "Terminating SSH server."
                kill `cat /etc/sshd.pid`
        fi
        ;;
*)
        echo "Usage: /etc/rc2.d/S99sshd { start | stop }"
        ;;
esac
exit 0
```

図 8: Solaris2.5.1 用 sshd 起動スクリプト例 (/etc/rc2.d/S99sshd)

3.4 ユーザ鍵の作成

次にユーザ鍵の作成を行ないます．先に述べたように，SSH を利用するためには認証用の RSA 鍵が必要です．計算機本体を認証するためのホスト鍵は，SSH インストール時に生成されています．利用者を認証するためのユーザ鍵は，利用者自身で用意する必要があります．利用者個人用の公開鍵および秘密鍵であるユーザ鍵を作成するコマンドは ssk-keygen です．特に指定しなければ，鍵ファイルは自分のホームディレクトリ下の ~/.ssh/ ディレクトリに保存されます．

```
wisdom% ssh-keygen ↵                                          ← 暗号鍵生成コマンド
Generating p:  ..............++ (distance 286)
Generating q:  ........................................++ (distance 762)
Computing the keys...
Testing the keys...
Key generation complete.
Enter file in which to save the key (/home/usr4/a70099a/.ssh/identity): ↵  ← 鍵ファイル置場の指定
Enter passphrase: ************************** ↵                  ← パスフレーズの入力
Enter the same passphrase again: ************************** ↵  ← パスフレーズの再入力
Your identification has been saved in /home/user/a70099a/.ssh/identity.
Your public key is:
1024 37 12565926471848073148614207696863271673408576073273987229346384576936664 5
11743397658519536773945898687779300072778394686104652806882416886810882227752518
87949220494005162007315160223595419144416936910534775093433232787723671409729894
63341796524112707855370823093190911051184920451049779410665375887480806221263 a7
0099a@wisdom
Your public key has been saved in /home/usr4/a70099a/.ssh/identity.pub
wisdom % ■
```

図 9: ユーザ鍵の生成

鍵の生成にはパスフレーズの入力が必要です．パスフレーズはパスワードのようなもので，公開鍵で暗号化された情報を復号化する際に秘密鍵とともに用いる情報です．生成した鍵を SSH で用いる人が，本当に鍵生成を行なった人であるのかどうかを認証するために用いられます．パスフレーズはファイルに記述されませんので忘れ

ないような文字列にして下さい[4]．

ここで注意して頂きたいのは，ただの **telnet** などで接続した遠隔地の計算機上で **ssh-keygen** の実行は絶対に行なうべきではない，という事です．遠隔利用中にパスフレーズを入力したならばパスフレーズ盗聴の可能性があります．パスフレーズが盗聴されれば，telnet におけるパスワード盗聴の場合と同様に，盗聴者が他者になりすまして SSH ログインできてしまうため，SSH を使う意味がありません．自分の手元にある計算機のコンソール上で鍵生成コマンドを入力するのは問題ありません．遠隔地の計算機上で鍵を生成する場合，例えば図 9 のような wisdom 上での鍵生成のような場合には，まず ssh を使ってホスト認証だけでログインして鍵生成を行なって下さい．

万一パスフレーズを忘れてしまった場合には，もう一度鍵を作成しなおして下さい．鍵の作成コマンド ssh-keygen コマンドを入力すると，鍵の作成が開始されます．SSH では 3.5 節で述べるような鍵の登録が必要です．鍵を再作成した場合は，作成しなおした鍵を再度登録しなおす必要があります．鍵ファイルの登録方法については，3.5 節および図 10 を参照して下さい．

3.5 SSH を用いた遠隔計算機利用例

ここでは「SSH を用いて wisdom から kyu-cc を遠隔利用する」という例を用いて，説明します．実際には利用者は外部からセンターの計算機を利用する事の方が多いでしょうから，wisdom を SSH クライアントとなる計算機，kyu-cc を SSH サーバとなる計算機として考えて下さい．

まず，前節で説明したように，SSH クライアントとなる wisdom 上で利用者各自のユーザ鍵を `ssh-keygen` コマンドを用いて生成します．次に wisdom 上で生成したユーザ鍵のうち，公開鍵であるファイル `identity.pub` を kyu-cc へ複製します．図 6 で説明したように，公開鍵はサーバ側で既知でなければならないからです．公開鍵ファイルの複製には ftp などを用いれば良いでしょう．wisdom 上の `identity.pub` ファイルを kyu-cc 側に複製した後，kyu-cc 上でファイル名を `authorized_keys` という名前に変更し，`~/.ssh/`ディレクトリに保存します．

SSH を用いて他の計算機から kyu-cc を利用した事がある場合には，既に `~/.ssh/`内に `authorized_keys` ファイルが存在しているでしょう．この場合は，既存の `authorized_keys` ファイルに wisdom の `identity.pub` ファイルの内容 (鍵データ) を追加記述して下さい．

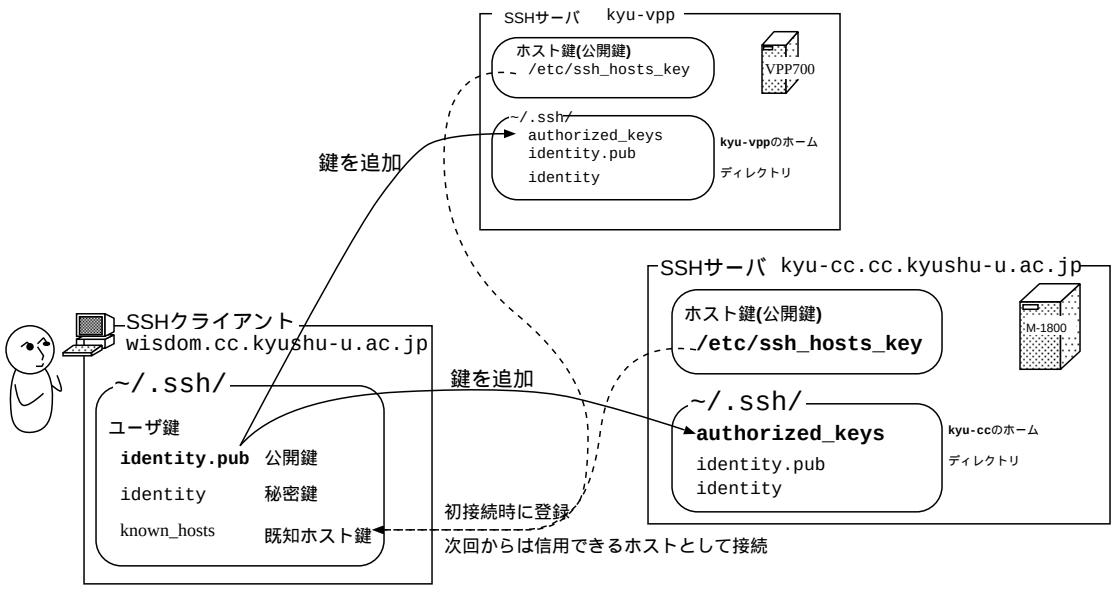

図 10: ユーザ鍵の追加

ユーザ公開鍵の交換が終了しましたら，SSH を用いて wisdom から kyu-cc へログインしてみましょう．コマ

[4]私は好きな歌のフレーズや言葉を使うようにしています．

ンドは ssh です．クライアント側とサーバ側のユーザ名が同じ場合は ssh の後に計算機を指定するだけで接続できます．ユーザ名が異なる場合，-l オプションでサーバ側のユーザ名を指定し，その後にサーバ計算機名を入力します．図 11 に ssh の入力例を示します．SSH を利用する際に，正当な人が利用しているのかを示すためにパスフレーズの入力が必要です．また，あるサーバ計算機に初めて接続する際には，その計算機のホスト鍵を登録するかどうかの質問が行なわれます．ここで yes を入力すると，クライアント側の known_hosts というファイルに，ホスト鍵データが保存されます．

```
wisdom % ssh -l a70099a kyu-cc.cc.kyushu-u.ac.jp ↵
Host key not found from the list of known hosts.                    初接続時のホスト鍵登録の確認質問
Are you sure you want to continue connecting (yes/no)? yes ↵        次回同ホスト接続時に質問は無い
Enter passphrase for RSA key 'itou@wisdom': ********************** ↵   パスフレーズ入力
Last login: Mon Mar 29 18:05:23 1999 from wisdom.cc.kyus
kyu-cc % ■
```

図 11: SSH でログイン

他に，ssh を経由して遠隔地の計算機からファイルを複製することもできます．この場合は，scp コマンドを用います．scp は複製をする cp コマンドと同じように使用する事ができ，rcp と同様に複製ファイル名の前に計算機名を指定します．

```
wisdom % scp a70099a@kyu-cc.cc.kyushu-u.ac.jp:~/my.txt ~/. ↵
Enter passphrase for RSA key 'itou@wisdom': ********************** ↵   パスフレーズ入力
my.txt               |         39 KB |   8.0 kB/s | ETA: 00:00:00 | 100%

wisdom % ■
```

図 12: scp によるファイルの複製

4 SSH の利用 (Windows95/98)

Windows95/98[5] 上で SSH を利用するためのソフトウェアがいくつか開発されています [6, 7, 8]．ここでは，その一つである TTSSH[9] を紹介します．TTSSH は，Windows95/98 で利用できる Tera Term(Pro)[10] を拡張し，SSH 利用できるようにするものです．Tera Term(Pro) は MS-Windows 用ターミナルエミュレーター (通信用ソフトウェア) です．VT100 エミュレーション，telnet 接続，シリアルポート接続 (パソコン通信) に対応しています．Tera Term と TTSSH の両方ともフリーのソフトウェアですので，センター利用者も使う事ができます．

4.1 インストール

2 つのソフトウェアのインストール方法を説明します．まず，ソフトウェアの入手先 URL を示します．1999 年 3 月 31 日現在，それぞれの最新バージョンは，Tera Term Pro ver.2.3 for Windows95/NT (ファイル名は tterm23.zip), TTSSH ver.1.4 (ファイル名は ttssh14.zip) です．

Tera Term Pro	http://hp.vector.co.jp/authors/VA002416/
TTSSH	http://www.zip.com.au/~roca/ttssh.html

それぞれのファイルを入手しましたら，展開ソフトウェアを用いてファイルを展開します[6]．Tera Term のインストールは，インストール用ソフトウェアがありますので簡単に行なえます．いわゆる DOS/V PC の場合，特

[5] Microsoft 社の Windows95 および Windows98 の事です．
[6] 私は展開用ソフトウェアとして lhasa を用いています．(http://www.digitalpad.co.jp/~takechin/)

に指定しなければ，C:￥Program Files￥Ttermpro￥にインストールされるでしょう．

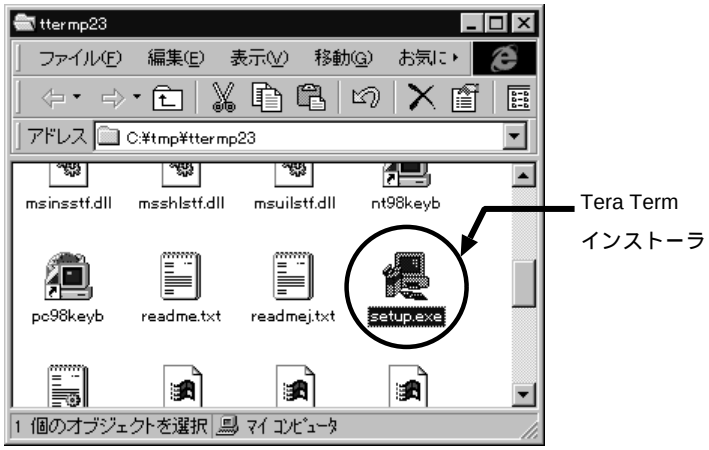

図 13: Tera Term インストールソフトウェア

TTSSH のインストールついては，TTSSH の WWW ページに記述してあるように作業を行ないます．まず ttssh14.zip ファイルを展開します．すると図 14 のように，Libeay.txt，libeay32.dll，Readme.txt，ttssh.exe，TTXSHH.dll という 5 つのファイルが生成されます．これらを Tera Term Pro のフォルダに移動してインストール作業は終了です．

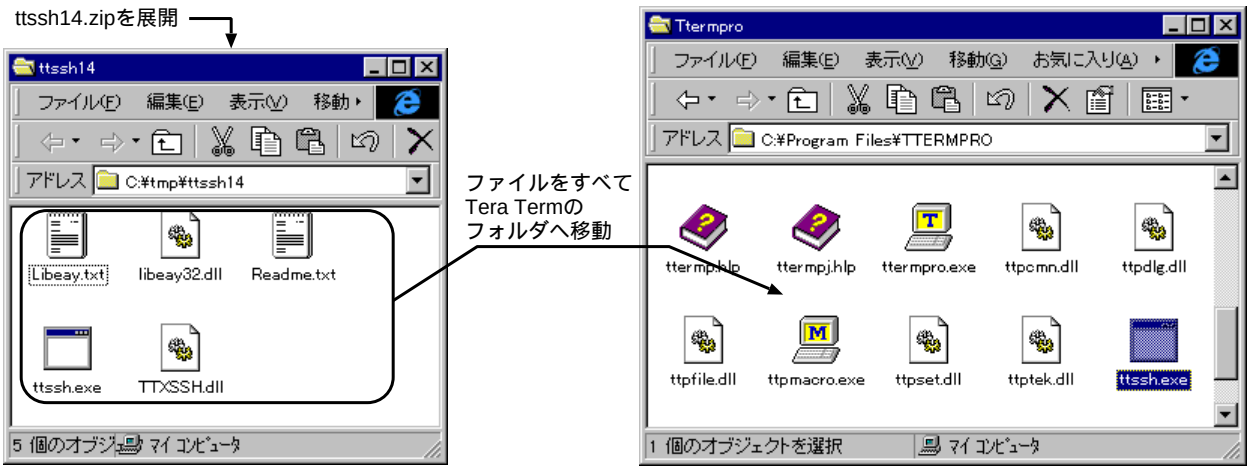

図 14: TTSSH のインストール

4.2 鍵ファイル

次に認証用の RSA 鍵を用意する必要があります．ユーザ鍵データを作成するためのコマンドは TTSSH には含まれていません．鍵作成には他の Windows 用 SSH ソフトウェアを使用する手もありますが，UNIX 計算機上で作成した鍵データをそのまま用いる方が便利でしょう．3.5 節の図 10 のように，wisdom 上で ssh-keygen コマンドを入力して利用者鍵を生成します．パスフレーズも必要です．

wisdom で利用する鍵と同じユーザ鍵を使うと混乱しそうなので，Windows 用に別の鍵を作成しましょう．鍵作成コマンドは同じ ssh-keygen で，鍵を保存するファイルの名前を指定することで，別のファイルとして鍵データを保存できます．

```
wisdom% ssh-keygen -f new.my.key ↵                    ← 暗号鍵生成コマンド (鍵ファイル名指定)
Initializing random number generator...
Generating p:  .....++ (distance 76)
Generating q:  ...........++ (distance 210)
Computing the keys...
Testing the keys...
Key generation complete.
Enter passphrase: ************************** ↵        ← パスフレーズの入力
Enter the same passphrase again: ************************** ↵   ← パスフレーズの再入力
Your identification has been saved in new.my.key.
Your public key is:
1024 37 135167431811817247045297455626463143377751240485105953763812398903342954
074070747938669176456754348191531649802346523333719546807081360895516111446947 90
070392835506325983141943362867861805222042917688136581716419117505890545156219 84
846678072577541482204708053701174608138021941668225626079918763200912598407 77 i7
0034a@wisdom                                  ← “wisdom” の所は後で自分の計算機名に変更する必要あり
Your public key has been saved in new.my.key.pub
wisdom % ■
```

図 15: ユーザ鍵の生成

上記の鍵は，wisdom 上で作成しているので最後の計算機名が wisdom になっています．計算機名の部分を，使用する Windows 計算機のインターネット上の名前に変更して下さい．次に ftp などを用いて自分の Windows 計算機に 2 つの鍵ファイルを持って来ます．

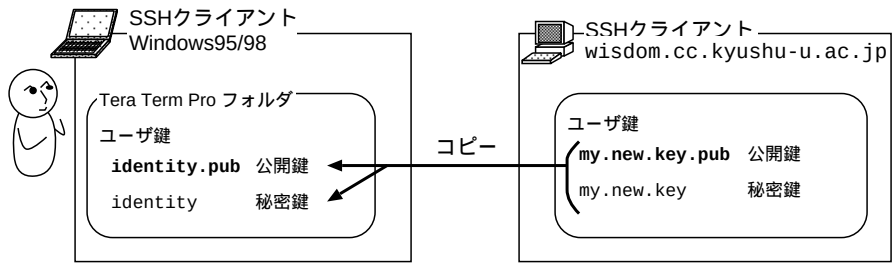

図 16: ユーザ鍵の複製

TTSSH でもユーザを確認するためのパスフレーズの入力が必要になります．図 16 のように，wisdom のユーザ鍵を利用する場合，wisdom での鍵生成時に入力したパスフレーズを入力する必要があります．鍵生成時に利用したパスフレーズを憶えておいて下さい．

4.3 実行

Tera Term と TTSSH をインストールしたフォルダにある `ttssh.exe` というファイルを実行すれば，TTSSH が起動します．もちろん Tera Term の `ttermpro.exe` を実行すれば，普通の Tera Term が起動します．図に TTSSH の起動画面を示します．

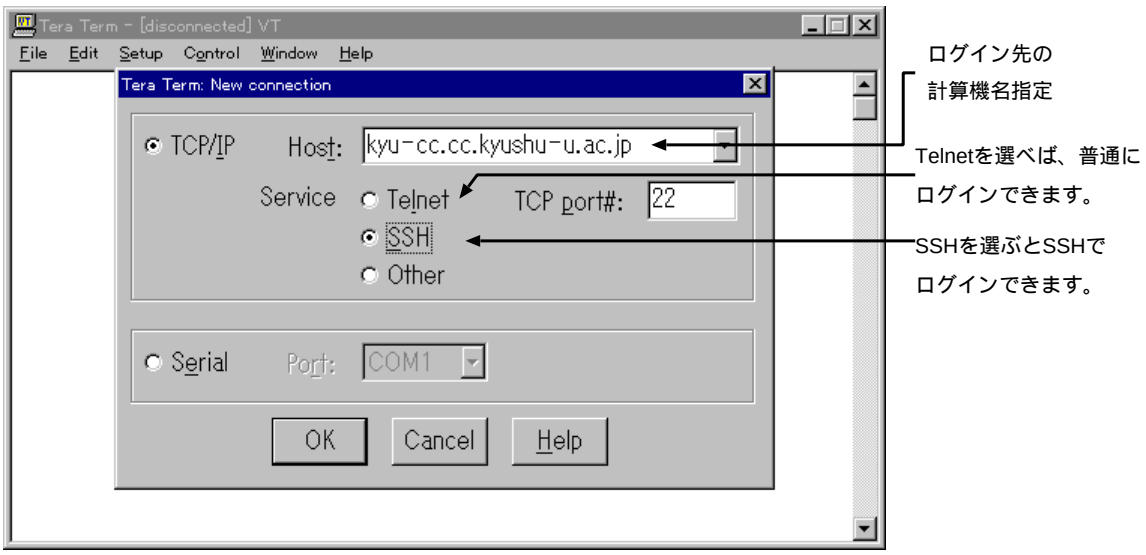

図 17: TTSSH の起動

4.4 TTSSH の設定

インストール直後に TTSSH を起動した場合，設定が不十分なため SSH の機能全てを利用できません．ユーザ鍵ファイルの名前，既知ホスト鍵を保存するためのファイル名などの設定をする必要があります．

起動時 (図 17) に Cancel ボタンを押して接続前のウィンドウを表示して下さい．[Setup] → [SSH...] とメニューを選択すると，図 18 左側のウィンドウが表示されます．これは暗号方式の選択と，既知ホスト鍵を保存するファイルを指定する設定画面です．[Setup] → [SSH Authentication...] とメニューを選択すると，ユーザ鍵を保存しているファイルの指定を行なう事ができます．これらの設定を行なってください．設定が終了しましたら，[Setup] → [Save setup...] を選択して，設定を保存して下さい．

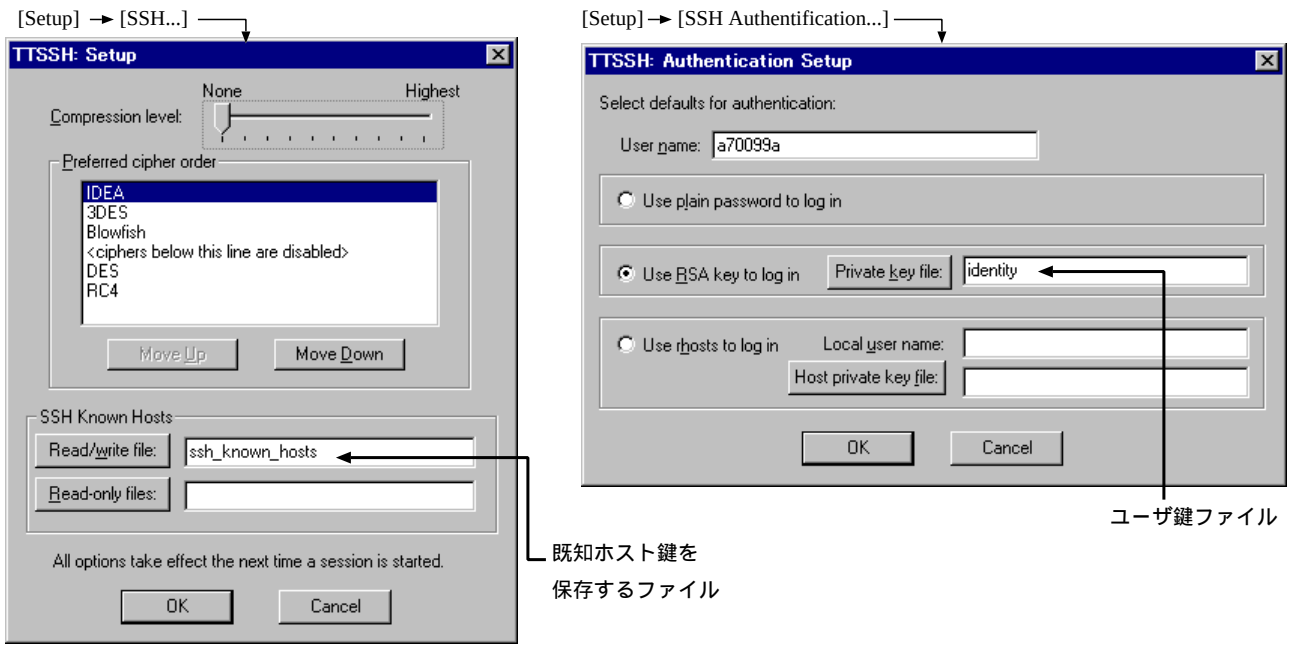

図 18: TTSSH 設定画面

5 おわりに

今回は通信内容を暗号化する SSH について解説しました．研究者だけのネットワークだったインターネットは，現在では企業や団体，個人へと利用者の裾野を広げ，誰にでも使える社会の情報基盤となりつつあります．ネットワークが使えるという利便さの反面，盗聴やなりすましなどの危険性が増えてきたことも事実です．そこで，危険を防いで安全にネットワークを利用するための技術が開発されつつあります．

インターネット上に悪意を持つ人の存在を仮定せずに，暗号化や認証などの防衛手段を使用しないという方法もあるでしょう．とはいうものの実世界では，お出掛け前に家に鍵を掛けない人は少ないと思います．つまり，ほとんどの人が泥棒などの悪意のある者の存在を仮定して行動しているわけです．インターネットも人間が構築して利用しているのですから，社会から解離した存在ではなく実社会の一部であり，悪意を持つ人も存在します[7]．

今回解説した SSH は，インターネットでの通信の安全性を確保するための技術の 1 つです．ただし SSH を用いても完全に安全という訳ではありません．例えば，個人の秘密鍵とパスフレーズを知られたならば，他人にログインされる可能性があります．それでも telnet と異なり，平文でパスフレーズが盗聴される可能性は随分と減少します．自分の事は自分で守れるような仕組を利用してみてはいかがでしょうか．

参考文献

[1] “IEEE 802 LAN/MAN Standards Committee”, http://grouper.ieee.org/groups/802/ , 1999 年 3 月 31 日現在.

[2] “SSH Home Page”, http://www.ssh.fi/, 1999 年 3 月 31 日現在.

[3] “IETF Secure Shell Working Group”, http://www.ietf.org/html.charters/secsh-charter.html , 1999 年 3 月 31 日現在.

[4] 太田和夫, 黒澤馨, 渡部治 : “情報セキュリティの科学 ~ マジックプロトコルへの招待 ~ ”, ブルーバックス B-1055, 講談社, 1995.

[5] “RSA Home Page”: http://www.rsa.com/ , 1999 年 3 月 31 日現在.

[6] 島 慶一 : “UNIX 知恵袋 Secure Shell”, UNIX マガジン 1998.6, pp.41-48.

[7] 島 慶一 : “UNIX 知恵袋 Secure Shell (2))”, UNIX マガジン 1998.7, pp.62-69.

[8] 島 慶一 : “UNIX 知恵袋 Secure Shell (3)”, UNIX マガジン 1998.8, pp.47-58.

[9] “TTSSH ホームページ”, http://www.zip.com.au/ roca/ttssh.html, 1999 年 3 月 31 日現在.

[10] “Tera Term (Pro) ホームページ”, “http://hp.vector.co.jp/authors/VA002416/ ,” 1999 年 3 月 31 日現在.

[7] この原稿を書きあげた後，筆者の WS が NFS を利用した方法で外部から侵入されました．危険は確実に存在します．