UNIXネットワークセキュリティー入門

只木進一*

1 はじめに

コンピュータネットワークは急速に私たちの身の周りで使われるようになっています。以前に、本広報に電子メールの紹介を書いた頃とは大きな違いです [1]。一般家庭までコンピュータが広がりました。そのため、ほんの一部の人が研究用に使っていた頃には予想もしなかったことが発生するようになりました。普通の社会で起こる不正が同じように起こるようになったのです。通信を盗聴したり、他人になりすましたりするような不正です。

もしもあなたがネットワークを利用しているならば、他人ごとではありません。例えば、電子メールを利用する際には、パスワードが必要です。そのパスワードは他で使っているパスワードと同じではありませんか。ネットワーク上を流れるパスワードを盗聴することは、その気になれば簡単です。ネットワークを流れるあなたのパスワードが、あなたの銀行口座のキャッシュカードで使っているものと同じパスワードならば、どんなことが起こるか考えてみましょう。

あるいは、あなたは十分にセキュリティーに注意を払っているかもしれません。しかし、あなたと同じコンピュータを利用している, あるいは同じネットワーク上にコンピュータを接続している他の人はそうではないかもしれません。そういうセキュリティーにルーズな人の ID やコンピュータから不正攻撃を企てる人は入り込み、通信を全て盗聴するかも知れません。他人のセキュリティーの甘さが自分に害を及ぼす可能性もあるのです。

さらに、あなたの ID やコンピュータが不正攻撃者に乗っ取られてしまうと、不正攻撃者はあなたになりすまして、他の利用者やコンピュータへ不正攻撃を仕掛けます。つまり、あなたは、不正攻撃者の協力者になってしまいます。

最近の不正攻撃の手法では、あるネットワーク内の全てのコンピュータに対して、そのコンピュータが持っている全てのサーバ機能を調べて、セキュリティー対策の行われていないものに攻撃を加えてきます。例えば、佐賀大学内にある個人的なメールサーバが、セキュリティー上の問題を持っていれば、それを見つけて攻撃をしかけてきます。つまり、ネットワークに情報機器を接続すれば、常に不正攻撃の対象になってしまいます。このような攻撃から身を守るにはどうしたら良いでしょうか。

近年、ワークステーションが低価格になると同時に、安定して稼働するパーソナルコンピュータ用 UNIX が普及し、UNIX を利用する人の数が増えてきました。本学の中にも、多数のパーソナルコンピュータ UNIX が稼動していることでしょう。UNIX は、複数の人間がネットワークを介して利用できるように設計されたシステムですから、ネットワークを介した不正攻撃の対象にもなり得ます。そこで本稿では、普通の人が UNIX を運用する場合のセキュリティー対策の基本について見て行きましょう。

2 一般論

まず、UNIX のネットワークの基本を概観して、セキュリティー対策の一般論を見て行きましょう。

インターネットで使われているプロトコルは TCP/IP と総称されています。通信内容は、パケットと呼ばれる小さな塊になって送られます。パケットには、郵便のように表書きがあり、発信元、送信先、サービス番号などが書かれています。サービス番号とは、電子メール、POP あるいは WWW などのサービスを区別するための符号です。不正攻撃はパケットの内容のほうですから、封を切らなければ良いのです。つま

*理工学部知能情報システム学科

り、セキュリティー対策の基本の一つは、このパケットの表書きを見て、受け取ったり、受け取りを拒否したりすることになります。

UNIX は複数の利用者が複数の作業 (タスクと呼ぶ) を実行することができる OS です。システムに常駐するタスクを daemon と呼びます。ネットワークからの接続要求を受けるのも daemon です。ネットワークサービスプロセスの起動方法には二つの方法があります。一つはそれ自身を daemon として常駐させる方法です。電子メールサーバがその例です。しかし、あまり接続要求の多くないサービスを常駐させるのは資源の浪費です。そこで、`inetd` という daemon を代表として常駐させ、接続要求内容に応じてそれぞれのプロセスを起動する方法が使われています。telnet サーバなどがその例です。

不正攻撃は、これらのネットワークサービスのセキュリティー上の弱点を狙って来ます。そこで、セキュリティー対策の基本の第二は、不要なサーバを起動させないことになります。

どんなに対策を講じても、セキュリティー上の弱点を全て無くすことはできません。また、今後、どんな新種の攻撃が出てくるかも分かりません。そこで、システムに起こった事柄を常に監視しておくのが、セキュリティー対策の基本の第三となります。ネットワークサービスへの接続が、何時、どこから行われたかを記録することです。

通常のネットワークを介した利用では、パスワードが平文、つまり暗号化処理なしにネットワークを流れます。特に管理者のパスワードが洩れた場合、システムに深刻な影響が出る恐れがあります。そこで、ネットワークを介したパスワード入力を極力行わない、どうしても必要な場合には暗号化をするのが第四の対策です。

3 UNIX のネットワーク関係の設定

ここでは、UNIX のネットワークセキュリティーに関した設定の方法を見て行きましょう。OS に依存する部分もありますので、ここではパーソナルコンピュータ用 UNIX の一つ FreeBSD3.3[2] を想定して進めていきます。

3.1 ネットワークサーバの確認

まず、不必要なサーバが起動していないかの確認をしましょう。パーソナルコンピュータ用 UNIX の場合は、インストール時に各種ネットワークサーバを起動するか否かを聞いてくるものがあります。本当にサーバを上げて管理する気が無いのならば、サーバ類は上げないようにしましょう。よく使われるサーバを表 1 に挙げておきます。

前述のように、サーバの起動方法には二つあります。一つは、単独の daemon として起動するもの、他の方法は、`inetd` という代表サーバを介して起動するものです。

単独の daemon として起動するものは、更にシステム標準のものと、後から追加するものに分けることができます。`sendmail` のようなシステム標準のものは、FreeBSD の場合は、`/etc/rc.conf` に起動の可否とオプションを記述します。FreeBSD3.3 の場合、システム標準のスクリプトが`/etc/defaults/rc.conf` にあり、そのスクリプトとの相違点だけを`/etc/rc.conf` に記述します。他の OS の場合、`/etc/rc.d` の下に起動スクリプトを置くものもあります。幾つかのシステム標準のサーバについて注意点を挙げておきましょう。

電子メールサーバ `sendmail` は標準で起動されるサーバの一つです。通常はオプション`-bd` を付けて起動します。この状態では、サーバは送信と受信を行います。電子メールは最も使われる機能であるため、最も頻繁に不正攻撃を受けます。そこで、もしもあなたの使っているコンピュータが電子メールの受け取りを直接にする必要が無いならば、このオプション`-bd` を外しましょう。この状態でも電子メールの送信は可能です。メールを読むのは、`pop` などで信頼できるサーバから行うこともできます。

NIS と NFS は、ネットワークを介してパスワードやファイルを共有するシステムです。パスワードファイルは暗号化されていますが、時間をかければ弱いパスワードは破られてしまいます。ファイル共有しているファイルの中に、パスワードを連想させるものが混じっていたり、システムセキュリティー関連の情報が入っていると、セキュリティーを脅かします。これらは、プライベートネットワークなど閉じたネットワーク内に限定するだけでなく、アクセスを許可する範囲を限定して使います。

表 1: よく使われるネットワークサーバ

サーバ名	説明	起動、設定など
ftpd	匿名ファイル転送を行う。通常のユーザ認証を伴うものとは異なる。	/etc/passwd にftp というユーザが作られると可能となる。サーバの起動はinetd から行われる。
sendmail	電子メールの配送送受信を行うサーバ。	起動は/etc/rc.conf に記述されている。通常は-bd というオプションを付けるが、メールを受け取らない場合には、このオプションを外す。
popd	pop プロトコルによるメール読み出しサーバ。	サーバの起動はinetd から行われる。
imapd	imapp プロトコルによるメール読み出しサーバ。	サーバの起動はinetd から行われる。
httpd	WWW のサーバ。	/usr/local/etc/rc.d/の下の起動スクリプトで起動する。
smbd	Windows とのファイル共有のためのサーバ。	サーバの起動はinetd から行われる。
inetd	ネットワークサーバの代表サーバ。	起動は/etc/rc.conf に記述されている。
xntpd	時計を合わせるサーバ。	起動は/etc/rc.conf に記述されている。

NIS や NFS を使う際に必要な daemon に portmap があります。これを使った不正攻撃も頻繁です。NIS や NFS を使わないならば/etc/rc.conf で対応する個所に NO と記述しましょう。

システムに後から追加するサーバ類の起動は、FreeBSD の場合には、/usr/local/etc/rc.d の下に起動スクリプトを置きます。スクリプトが実行可能なようにファイルパーミッションを設定することに注意します。他の OS では、/etc/rc.local などのスクリプト内に対応するスクリプトを記述するものや、/etc/rc.d などのディレクトリ内にスクリプトを置くものなどもあります。幾つかのサーバについて、注意点を挙げておきましょう。

電子メールを Windows などから読み書きするプロトコルに pop や imap があります。これらのサーバは、通常は inetd から起動します。これらのプロトコルの通常の利用では、平文、つまり暗号化なしにパスワードを頻繁に流します。このため、情報処理センターでは、学外との通信を止めています。利用する場合には、ごく近くのネットワークからだけ使えるような処置が必要です。

Windows とのファイル共有を行う smbd も便利なサーバの一つです。NFS の場合と同様にセキュリティーの弱点となるので、設定ファイルでアクセス制限をかけます。

3.2 アクセス記録の保持

セキュリティー保護の基本中の基本は、サーバへのアクセスの記録を残すことです。サーバの設定でどのようにセキュリティー保護をしても、それが期待通りに動いているか否かを見なければなりません。また、新手の不正攻撃があるかもしれません。そこで、サーバ類はアクセスの記録を残すものを使う必要があります。もしも、OS 標準のサーバがアクセス記録を残さないならば、記録するように設定を変更するか、記録できるサーバと入れ替える必要があります。

UNIX の場合、システムの記録を残す syslogd という daemon が常駐し、その機能を使って記録を残すものが多くあります。どのような機能と状況がどのファイルに記録されるかは、/etc/syslog.conf に定義されています。この設定を変更した場合は、syslogd へ

HUP というシグナルを送るか、システムを再起動します。syslogd は出力先のファイルを自分で生成することはしませんから、新しい出力先ファイルを定義した場合は、空のファイルを手で作っておきます。

システムの記録が多くなりすぎて、ファイルシステムを圧迫する危険もあります。UNIX の場合、システムの記録を定期的に圧縮する仕組みがあります。FreeBSD の場合、記録を記述しているファイルが一定サイズ以上になると、圧縮して、適当な期間だけ保持する仕組みがあります。設定は/etc/newsyslog.conf で行います。

sendmail は、基本の設定のままで記録を残します。FreeBSD の標準では、/var/log/maillog に記録します。

図 1: /etc/syslog.conf への inetd ログの追加。/etc/syslog.conf では、項目の区切りはタブです。

```
auth.info       /var/log/info
```

一方、inetd は、標準では記録を残さない daemon です。しかし、FreeBSD3.3 で使われている inetd は後述する tcp_wrapper の機能が組み込まれ、アクセス記録を残すことが出来ます。標準の設定のままでも、アクセスを拒否した場合の記録が/var/log/messages に残されます。アクセスを許可した場合の記録を残すには、inetd のオプションを標準の-wW から-lwW に変更します。設定変更は、/etc/rc.conf で行います。更に、/etc/syslog.conf に図 1 の行を加えて、システムを再起動します。すると、全ての記録が/var/log/info へ記録されます。

システムに後から追加するサーバ類は、通常、アクセス記録を残す機能がついています。多くは、syslog の機能ではなく独自に記録します。どこに記録するかは、設定ファイルで指定します。

3.3 アクセス制限

ネットワークサーバには、httpd のように、アクセス元が特定されないものと、telnetd のように、特定マシンからしかアクセスしないものがあります。このように、サーバごとにアクセス元に制限を行うことが賢明です。

httpd は smbd のように、設定ファイルでアクセス元を特定できる場合には、設定します。特に、smbd のようなファイル共有の場合には、不必要にアクセス制限を緩めてはいけません。

inetd の場合、従来はアクセス制限を行うことは出来ませんでした。そこで出てきたのが tcp_wrapper です [3]。従来の使い方では、/etc/inetd.conf を書き換えて、inetd が受け取ったリクエストを一旦 tcp_wrapper に渡し、更に本来のプロセスに渡しました。この際に、記録が残り、制限リスト (/etc/hosts.allow 及び/etc/hosts.deny) に従ってアクセス制限が行われます [4]。

上述のように、FreeBSD3.3 で使われている inetd は tcp_wrapper の機能が組み込まれているため、/etc/inetd.conf を書き換えずにアクセス制限を制限リストを使って行うことが可能となっています。

sendmail も version.8.9 では tcp_wrapper を組み込むことができます。FreeBSD3.3 の場合は、既に組み込まれています。従って、sendmail.cf ではなく、/etc/hosts.{allow,deny}を使ってメールサーバへのアクセスを制限することができます。

3.4 通信の暗号化

ネットワークを介してリモートシステムを利用する場合を考えましょう。パスワードを入力すると、そのパスワードは平文、つまり暗号化されずに流れます。パスワードがもれてしまうと、いくらアクセス制限をしてもその効果は大きく下がってしまいます。リモートシステム利用に際して良く利用される暗号化技術の一つが Secure Shell (ssh) です [5]。前回 [4] にも書きましたが、アメリカ合州国の輸出規制があるので、対応するソフトウェアをアメリカ合州国からダウンロードすることは違法です。

前回の解説時には、ssh-1 しかありませんでしたが、現在のバージョンは ssh-2.0.13 です。ssh-1 と ssh-2 には互換性が無く、更に Windows 用クライアントが ssh-1 対応です。そこで、今回は両方を入れる方法について見て行きましょう。

まず、ssh-1 をインストールします。ソースを展開し、そのディレクトリで

```
configure
```

図 2: /etc/ssh2/sshd2_config のシステム機能の変更

```
#       SyslogFacility                  AUTH
        SyslogFacility                  LOCAL7
```

図 3: /etc/ssh2/sshd2_config へのsshd-1 互換の記述

```
        Ssh1Compatibility               yes
        Sshd1Path                       /usr/local/sbin/sshd1
```

```
make
```

を実行することでコンパイルできます。後は、make install でシステム領域へインストールします。この時、設定ファイルが/etc の下に書かれます。そのうちの、/etc/sshd_config の syslog 機能の部分だけを編集しておきましょう (図 4)。ssh では、この図のように、設定ファイルで syslog のどの機能を使うかを指定できます。ここでは LOCAL7 を使うことにしましょう。

図 4: /etc/sshd_config のシスログ機能の変更

```
SyslogFacility LOCAL7
```

次に、ssh-2 もコンパイル及びインストールをします。コンパイル方法は、ssh-1 と同様です。インストールすると、設定ファイルが/etc/ssh2 の下に作られます。ここでも、/etc/ssh2/sshd2_config を編集しましょう。まず、syslog 機能を LOCAL7 を使うようにします (図 2)。

次に、ssh-1 との整合を取る部分を編集します (図 3)。この機能によって、もしもクライアントが ssh-1 対応であれば、sshd1 が起動されます。ついでに、/etc/ssh2/ssh2_config も書き換えて、通信相手が ssh-1 の場合に対応できるようにしましょう (図 7)。

図 5: /etc/syslog.conf へのsshd のログ機能の追加

```
local7.*        /var/log/sshd.log
```

上述のように、syslog に対して、LOCAL7 として記録しますから、対応する記述を/etc/syslog.conf に追加します (図 5)。対応するログファイルの定期的な更新を/etc/newsyslog.conf に書くのを忘れないでください。最後に起動スクリプト sshd.sh を /usr/local/etc/rc.d の下に置いて (図 6)、システムを再起動します。

図 6: /usr/local/etc/rc.d/sshd.sh

```
#!/bin/sh
sshd=/usr/local/sbin/sshd
if [ -f $sshd ]; then
        $sshd
        echo -n " sshd "
fi
```

次に、個人の設定を行います。ssh-keygen を実行すると、パスフレーズを聞いて来ます (図 8)。パスフレーズは、その名の通り、文章などを使います。パスフレーズを正しく入力すると、個人用の鍵が~/.ssh2/id_dsa_1024_a が作られ、同時に公開鍵 ~/.ssh2/id_dsa_1024_a.pub も作られます。そこで、~/.ssh2/identification に

```
IdKey id_dsa_1024_a
```

と記述し、自分の鍵を登録します。

このホスト (host.domain) に外部から ssh でリモートログインすると、通信は暗号化されますが、通常のパスワードでの認証が行われます。パスフレーズでの認証を行うためには、アクセス元のホスト (remote.domain) の id_dsa_1024_a.pub をアクセス先 host.domain へ~/.ssh2/remote.domain.pub としてコピーし、~/.ssh2/authorization に

```
Key remote.domain.pub
```

という行を記入します。

図 7: /etc/ssh2/ssh2_config へのssh-1 互換の記述

```
        Ssh1Compatibility               yes
        Ssh1Path                        /usr/local/bi/ssh1
```

図 8: ssh-keygen の実行

```
someone@host[1] ssh-keygen
Generating 1024-bit dsa key pair
   1 oOo.o
Key generated.
1024-bit dsa, someone@host.domain, Sat Jan 1 2000 00:00:00 +0900
Passphrase :
Again      :
Private key saved to /home/someone/.ssh2/id_dsa_1024_a
Public key saved to /home/someone/.ssh2/id_dsa_1024_a.pub
```

ssh は UNIX のシステムの標準である rsh と同様の使い方が可能です。

ssh ホスト名 リモートコマンド

という形で利用することができます。リモートコマンドの実行結果も暗号化されて戻って来ます。また、

slogin ホスト名

でリモートシステムに暗号化してログインしたり、

sftp ホスト名

で暗号化して ftp をかけることも可能です。

4 おわりに

本稿では、UNIX マシンをネットワークに接続する場合の、セキュリティーに関連した設定を簡単にまとめました。しかし、このような安全対策を継続して行っていくのは、非常に骨の折れる作業です。コンピュータ管理を本職としない、教職員にとっては尚更です。ネットワークに接続しないのが最も安全ですが、コンピュータネットワークの利便性から考えて、そうもいきません。そこで、サーバ機能を持たせる必要のない機器に関しては、外部から接続できないネットワークに接続するのが最も効果的な対策です。つまり、ファイアウォールなどを設置して、その内側に接続することです。

研究室や学科単位で、小さなネットワークを構築し、その内部でだけ有効なアドレス (プライベートアドレスと呼ぶ) を利用すれば、外部から不正攻撃されることを心配する必要はなくなります。外部と接続する必要がある場合だけ、インターネットアドレスが動的に付与される仕組みもあります。このような仕組みを NAT を呼びます。現在では、NAT の専用機が数万円で購入できます。ぜひ検討してみては如何でしょうか。

参考文献

[1] 只木進一「電子メールをお使いですか」、佐賀大学情報処理センター広報 No. **1** (1991) pp. 83 - 90.

[2] http://www.freebsd.org

[3] http://www.cert.org/ftp/tools/tcp_wrappers/tcp_wrappers_7.6.tar.gz

[4] 只木進一「あなたのコンピュータは大丈夫ですか」、佐賀大学情報処理センター広報 No. **8** (1999) pp. 1 - 9.

[5] http://www.ssh.fi