実践 Linux セキュリティー講座

今回はFTPとBINDについて取り上げます。SOHO環境ではFTPとBINDは必須というものではありませんが、よく使われるうえに、セキュリティーホールとなる可能性が高いものです。こうしたサービスを運用する場合には十分な注意を要するのはもちろんですが、できれば他のサービスで代用するか、業者に委託できないかといったことを検討してみるのも大事なことです。

第15回 FTPとBINDについて

ソフトウェアコンサルタント　すずきひろのぶ

FTPサーバーの役割

FTPサーバーはご存じのとおり、コンピュータ間でファイルを転送するためのサーバーです。FTPのが何のために用意されているのかを書き出してみると、多くのサイトでは次のようになるでしょう。

❶特定ユーザーがサイト内のマシン間でファイルを転送するため
❷特定ユーザーがインターネットを経由してファイルを転送するため
❸不特定多数のユーザーにAnonymous FTPのサービスを提供するため

❶に関しては、サイト内のマシンでファイルを転送する範囲ならば、リーズナブルな使い方だと思います。ただし、確実に内部の許されたマシンからのアクセスのみを許すように、tcp_wrapperを使ってアクセス制御をしてください。tcp_wrapperを使ってのアクセス制御については、本連載の1999年10月号で説明していますので、ここでの繰り返しの説明は避けます。すでにtcp_wrapperを設定している場合には、現在正しく設定されているかをチェックしてみてください（図❶）。

外部からのアクセスは極力避ける

❷の「特定ユーザーがインターネットを経由してファイルを転送するため」のケースは、ウェブサーバー上のコンテンツを外部に委託している企業や組織に見られる運用方法です。プロバイダーのウェブサーバー上にある自分のウェブページを更新する際などにもFTPを使うことがほとんどです。

SOHO環境などでは、仕事上のファイルのやり取りに使うケースが多いでしょうが、こうした用途でFTPサーバーを使うのは、とかくずさんな運用になりやすいため、あまりおすすめできません。どうしてもFTPサーバーを使う必要があるなら、次のことを厳守してください。

i）FTPのアカウントとパスワードの管理をしっかり行う。adminやmanagerといったアカウント名、あるいは簡単に見つけられるパスワードを絶対に使わない。また、定期的にパスワードを更新する。

ii）接続相手のIPアドレスをtcp_wrapperのALLOWのルールに加え、それ以外の接続は認めない。

それであっても外部からの接続には「パスワードが盗聴される」という危険性があります。こうした危険を回避するためにはSSHを利用するのがいいでしょう。SSHにはサーバーへの接続やファイル転送を暗号化して行う「sftp」というコマンドがあります（SSHの解説は1999年12月号と2000年1月号を参照して下さい）。

Anonymous FTPはウェブサーバーで代用

Anonymous FTPは不特定多数にファイルを提供するために用いられる方法ですが、そのことだけを考えると、Anonymous FTPはすでに役目を終えた方法なのかもしれません。ファイルを提供するということであれば、現在はウェブサーバーで十分だからです。

ミラーサイトを運用するような場合では、以前はファイルの同期をとるためのツールが「ftpmirror」というツールぐらいしかなかったので、FTPサーバーを使わざる得ないという状況もあったでしょう。しかしファイルの同期にしても、現在ではwgetというウェブサーバー上のファイルをまるごとコピーするプログラムがあるので、このミラーサイト問題も大きな障害にはならないと思います。

FTPサーバーは脆弱性の指摘が多い

FTPサーバーの問題は、何度となくプログラムのセキュリティーホールの指摘がなされていることです。過去3年間の「CERT Advisoryドキュメント」 Jump01 には、複数のFTPの脆弱性に関する警告が出ています。日本語の情報としては「Linuxのセキュリティー情報」 Jump02 が参考になります。

外部に公開して不特定多数がアクセスできるFTPサーバーを持つ場合は、頻繁にFTPの新しいセキュリティーホールが発見されていないか情報収集しておく必要が出てきます。常に情報収集することは、どんな場合も重要なポイントです。しかし、似たような機能のサーバーをいくつもメンテナンスしていく必要があるということを考慮すると、あまりおすすめできるようなものではないと思われます。他のサーバーで問題なく代替できる場合は、なるべく一本化して管理の手間をできるだけ少なくするほうが得策ではないでしょうか。メンテナンスができないなら初めから手をつけないというのも1つの大切な判断だと言えるでしょう。

Jump01 www.cert.org/ftp/cert_advisories/
※参照文書
CA-99-13-wuftpd.txt
CA-99-03-FTP-Buffer-Overflows
CA-97.27.FTP_bounce
CA-97.16.ftpd

Jump02 www.linux.or.jp/security/

FTPのアクセス制御を使う

FTPはtcp_wrapperによってアクセス制御を受けていますが（図❷）、FTP自身もアクセス制御の機能を持っています。ここでは、FTPの設定ファイルである、/etc/ftpaccess、/etc/ftphosts、/etc/ftpusersの記述を変えることで可能となる、簡単なFTPのアクセス制御について説明します。

Anonymous FTPを使わない

Anonymous FTPを許可せず、アカウントを持っているユーザーのみにFTPを許す方法です。/etc/ftpaccess中のクラスの定義からguestとanonymousを削除します（図❸）。

アカウントとホストによる制御

FTPサーバーにアクセスできるアカウントとホストを指定する方法です。/etc/ftphosts中でallowとdenyの定義を行います（図❹）。

アカウントによる制御

FTPにアクセスを許可しないアカウントの定義を行う方法です。/etc/ftpusersに、FTPに

図❶　tcp_wrapperが正しく設定されているかの再確認

```
%  /usr/sbin/tcpdchk -v
Using network configuration file: /etc/inetd.conf

>>> Rule /etc/hosts.allow line 7:
daemons:  in.telnetd
clients:  LOCAL 192.168.1.10
access:   granted

>>> Rule /etc/hosts.allow line 8:
daemons:  in.ftpd
clients:  LOCAL 192.168.1.   内側のネットワークユーザーのみアクセス可
access:   granted

>>> Rule /etc/hosts.allow line 9:
daemons:  ALL
clients:  ALL
option:   DENY
access:   denied
```

SSH（エス・エス・エイチ／Secure SHell）：ネットワーク上の通信内容を暗号化するプロトコル。通常のTELNETやFTPといったプロトコルでは、パスワードが暗号化されないままネットワーク上に流れてしまうといった危険性がある。

図❷ tcp_wrapperによるアクセス制御

/etc/inetd.confに以下の記述があれば、ftpはtcp_wrapperによってアクセス制御がかけられている。

```
ftp       stream    tcp       nowait    root      /usr/sbin/tcpd      in.ftpd -l -a
```

図❸ Anonymous ftpを拒否する

/etc/ftpaccessのクラスの定義から、guestとanonymousを削除する

変更前

```
class   all   real,guest,anonymous *
```

変更後

```
class   all   real *
```

図❹ アカウントとホストで制御する

/etc/ftphostsに、allowとdenyの定義を追加する

```
allow hironobu 192.168.*.*——— ❶
deny * *—————————————————— ❷
```

❶アカウント名 hironobu に対し192.168.xxx.xxxのIPアドレス（ローカルアドレス）からのアクセスを許可する。
❷（allowで許可したユーザー以外の）すべてのftpでのアクセスを禁止する。

図❺ アカウントで制御する

/etc/ftpusersに、ftpでのアクセスを拒否するアカウントを追加する。

```
root
bin
......
nobody
hironobu
```

hironobu ——これでhironobuはftpでのアクセスが拒否され

図❻ /var/log/xferlog -- ftpの転送ログ

```
Mon Jan 17 13:18:30 2000 1 win 121 /tmp/x a _ i r hironobu ftp 0 *
Tue Jan 25 17:48:27 2000 1 pc 724 /tmp/index.html b _ i r hironobu ftp 0 *
```

（下線部の番号：「Tue Jan 25 17:48:27 2000」❶、「1」❷、「pc」❸、「724」❹、「/tmp/index.html」❺、「b _ i」❻、「r hironobu」❼、「ftp 0」❽）

❶時間 ❷転送にかかった時間 ❸相手ホスト名 ❹ファイルサイズ ❺ファイル名 ❻転送タイプなど ❼ユーザー名 ❽サービス名など

図❼ ftpwhoによる監視

```
# ftpwho
Service class all:
23150  ?  S    0:00 ftpd: localhost:
hironobu: IDLE
    -   1 users (no maximum)
```

※rootで実行する

サーバーに潜むバッファーオーバーフローの危険性

サーバーの脆弱性で一番厄介なのがバッファーオーバーフローの問題です。バッファーオーバーフローとは、プログラムの中で入力されるデータ長を考えていなかったために発生するバグのことです。このバグを悪用すると、外部から任意のプログラムを実行させることができます。たとえばDNSサーバーにバッファーオーバーフローを起こさせるようなデータを送り付け、シェルを起動させるような手法です。DNSサーバー（BIND）はroot権限で実行されますので、侵入＝root権限詐取となります。

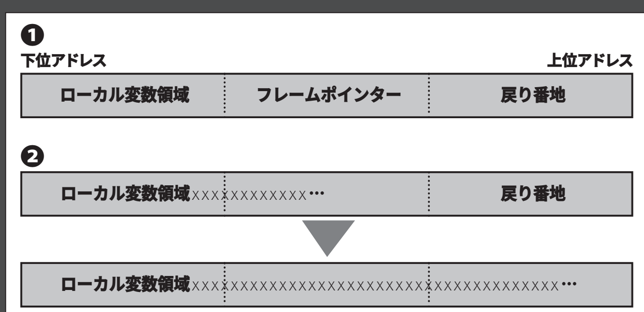
バッファーオーバーフローは技術的には以下のようなしくみになります。

CやC++などで書かれたプログラムでは、関数が呼ばれたときはそのローカル変数はスタック上に確保されます。関数が終了した時点で、スタック上のローカル変数領域やフレームポインターは開放され、制御の流れは戻り番地の値を見て関数が呼ばれた場所に戻っていきます（❶）。このローカル変数領域に対し、領域以上のサイズのデータを書いてしまうようなことができたらどうなるでしょうか？（❷）

不注意なプログラムではよく見られるバグで、通常はプログラムが異常終了してしまいますが、意図的に戻り番地の部分に偽の戻り番地を入れると、実際にその偽戻り番地を実行します。そこで、戻り番地の所にシェルコードを入るようにローカル変数領域を溢れさせます。この状態で、この関数を終了するとサーバープログラムがシェルに切り替わってしまいます。

このようにパスワードも何もなくてもシステムにログインした状態を作りだすことができます。この攻撃の恐い所はパケットのフィルタリング手法を使ったファイアーウォールは無効であるという部分です。「ファイアーウォールのシステムさえ入れて、あとはその後ろにシステムを隠せば十分だ」と思っている人もいるようですが、きちんとバッファーオーバーフローの対策が取られていないシステムをファイアーウォールの内側に入れてしまうと、わざわざ外部から内部へ侵入する入口を作ったようなことになってしまいます。

バッファーオーバーフローの本質的な問題はCやC++の「スタック上にバッファーを取る」といったメカニズム自体です。そのメカニズムがある限り、この問題は存在し続けることになります。



CERT（サート／Computer Emergency Response Team）：インターネットなどのネットワークを利用した不正アクセスへの対策を講じる団体。日本にはJPCERT/CC（コンピュータ緊急対策センター）がある。

よるアクセスを許可しないアカウントを記述します（図❺）。

FTPサーバーを運用する場合には、アクセスログも必ず確認してください。FTPの転送ログは/var/log/xferlogに記録されています。誰がいつ、どんなファイルをどこから転送したのかわかります（図❻）。また、FTPサーバーの現在の状態を確認するには、「ftpwho」というコマンドを使います（図❼）。

DNSとBINDとは

Linuxで DNS（ Domain Name System）の機能を提供するのがBIND Jump です。BINDはDNSとして必要なソフトウェア群を指します。実際にDNSとして実行されているプログラムはnamedという名前です。

DNSは主にドメイン名（ホスト名）からIPアドレスを検索するという重要な役目のために使われる機能です。DNSはホスト名に関する情報を管理するホスト情報分散データベースだと言えます。インターネット全体をカバーするために巨大な木構造のようなリンク関係を持っています。自分の管理するドメインのホスト情報をローカルに管理していますが、DNSを通して世界中に通知することになります（図❽）。

Jump www.isc.org/products/BIND/

DNSは管理が難しい

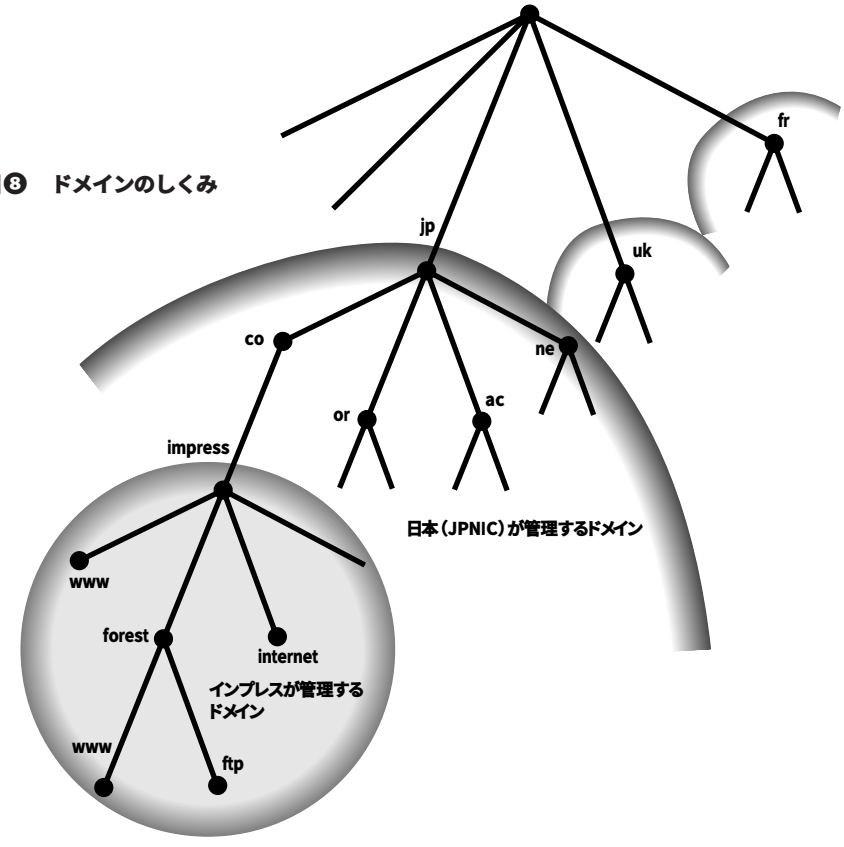
プロバイダーの提供するドメインのサブドメインを利用する場合は、DNSは自分で持たずにプロバイダー側で管理してもらうケースが多いと言えます。たとえばOCNユーザーの場合、OCNのサブドメインとして申請するとxxx-unet.ocn.ne.jpといった名称（xxxの部分はユーザーが決めることができる）が割り当てられ、この分のDNS管理はOCN側で行います。

xxx.co.jp（xxxはユーザーが取得したドメイン名が入ります）といった独自のドメインでも管理をプロバイダーが替わって管理するサービスが提供されている場合が多いようです。

正しくDNSを動かすには、DNSメカニズムの理解、ホスト定義の記述などの多くの知識が必要です。またDNSサービスが停止すると、インターネット上からそのドメインを見つけられないという致命的な問題を引き起こします。ですから信頼性の高い長期間安定して動作するマシン上で動作させる必要があります。そのような理由からDNSは多重にすることが望ましいので、プライマリーDNS（主DNS）だけでなく、別のマシンでセカンダリーDNS（予備のDNS）も用意する必要が出て来ます。

セキュリティー以前の問題としてこのような管理は大変です。またDNSがうまく設定されて動いていないと、ネットワーク回線は正常でもドメイン側へアクセスするのにいろいろな問題を引き起こしてしまいます。プロバイダー側としても慣れないユーザーに任せるより自分の所で管理したほうがサポートを考えるとスムーズに問題が解決するので積極的にDNSの肩代わりを提供しているものと思われます。

図❽　ドメインのしくみ



BINDもよく脆弱性が発見される

DNSサーバー環境を提供するBIND（named）はセキュリティーホールが見つかると深刻な問題を引き起こしやすいソフトウェアです。その見つかる頻度は低いとは言え、外部に直接接していること、そしてインターネット上の重要な管理ソフトであるということから、影響は大きいと言えるでしょう。そのため常にBINDのセキュリティー情報に目を光らせておく必要があります。これらを考えるとプロバイダーでDNSサービスをしてくれるのなら、そこに依頼して管理の手間を少しでも少なくする方が得策ではないでしょうか。どうしても自分のサイト上で管理する必要がある時のみ自分で持つのがいいでしょう。

次回は、境界ネットとインターネットを接続するルーターの設定についてお話したいと思います。

BIND（バインド／Berkeley Internet Name Domain）：カリフォルニア大学バークレー校（UCB）で開発されたDNSサービスのためのプログラムの名称。インターネットで用いられているDNSサービスの多くは、BINDかBINDを元に開発されたプログラムで運用されている。

［インターネットマガジン　バックナンバーアーカイブ］ ご利用上の注意

このPDFファイルは、株式会社インプレスR&D（株式会社インプレスから分割）が1994年～2006年まで発行した月刊誌『インターネットマガジン』の誌面をPDF化し、「インターネットマガジン　バックナンバーアーカイブ」として以下のウェブサイト「All-in-One INTERNET magazine 2.0」で公開しているものです。

http://i.impressRD.jp/bn

このファイルをご利用いただくにあたり、下記の注意事項を必ずお読みください。

- 記載されている内容（技術解説、URL、団体・企業名、商品名、価格、プレゼント募集、アンケートなど）は発行当時のものです。
- 収録されている内容は著作権法上の保護を受けています。著作権はそれぞれの記事の著作者（執筆者、写真の撮影者、イラストの作成者、編集部など）が保持しています。
- 著作者から許諾が得られなかった著作物は収録されていない場合があります。
- このファイルやその内容を改変したり、商用を目的として再利用することはできません。あくまで個人や企業の非商用利用での閲覧、複製、送信に限られます。
- 収録されている内容を何らかの媒体に引用としてご利用する際は、出典として媒体名および月号、該当ページ番号、発行元（株式会社インプレスR&D）、コピーライトなどの情報をご明記ください。
- オリジナルの雑誌の発行時点では、株式会社インプレス R&D（当時は株式会社インプレス）と著作権者は内容が正確なものであるように最大限に努めましたが、すべての情報が完全に正確であることは保証できません。このファイルの内容に起因する直接的および間接的な損害に対して、一切の責任を負いません。お客様個人の責任においてご利用ください。

このファイルに関するお問い合わせ先
株式会社インプレスR&D
All-in-One INTERNET magazine 編集部
im-info@impress.co.jp