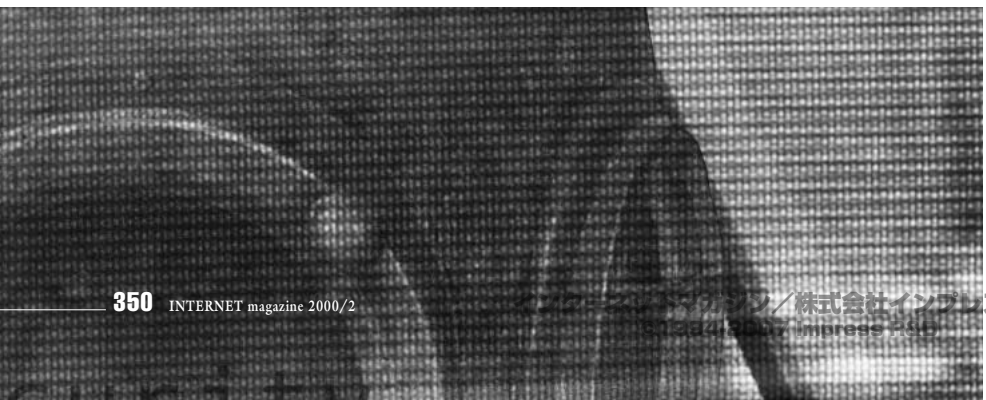
実践 Linux セキュリティー講座

今回はウェブサーバーのセキュリティーについて取り上げます。誌面の都合もありますので、具体的な設定は次回に説明するとして、今回は何がポイントなのかという概観的な話を中心に進めていきたいと思います。本連載ではEコマースを行うウェブサーバーのセキュリティーに関しては取り上げないことにします。また、ここでのウェブサーバーはApache（RedHatに付属のhttpサーバー）を前提にしています。

第13回 ウェブサーバーのセキュリティー（前編）

ソフトウェアコンサルタント　すずきひろのぶ



ウェブサーバーでの脅威

SOHOレベル、大組織のいずれにおいても、ウェブページが改ざんされることによる、もっとも深刻な「被害」は、実は信用毀損（信頼を損ねること）だと思われます。

たとえばビジネスを行っているサイトでウェブページの書き換えが一度でも発生すれば、ユーザーは誰もそのウェブサイトを信用せず、その企業と取引をする気はなくなるかもしれません。いずれにしても、企業としての信用度はガタ落ちと言えるでしょう。

どこからでも問題は発生する

ウェブサーバーにおける脅威には、いくつかのパターンがあります。ウェブで公開している情報の書き換え、ウェブサーバーにアクセスすると別のウェブサーバーに接続してしまう（ウェブサーバーの乗っ取り）、ウェブサーバー経由で非公開のファイルをコピーされてしまう、などが挙げられます。

これらはウェブサーバーのセキュリティーに直接の原因があるもの、それ以外の部分に起因し、ウェブサーバーのセキュリティーに影響を与えるもの、あるいはウェブサーバーのセキュリティーとは関係ない部分で発生しているものなどさまざまです。

運用ポリシーを明確に

ウェブサーバーの運用ポリシーが不明確であるために、セキュリティーに問題が発生している例がよく見受けられます。

たとえば、外部から直接リンクを張ってはいないものの、ウェブサーバーの公開ディレクトリー上にあるファイルを、外部の第三者にアクセスされてしまうような場合です。これは正確には不正アクセスではありません。アクセスできるファイルにアクセスされたにすぎないのです。

過去に雑誌などで取り上げられた例として、

Apache（アパッチ）：世界中のボランティアによって開発が続けられているフリーのHTTPサーバーソフト。Linux、FreeBSDのほか、ウィンドウズにも移植されている。SSLなど機能をモジュール単位で追加できるのが特徴。

ウェブから登録された個人情報ファイルにウェブ経由でアクセスされてしまったというものがあります。詳細は公開されていませんが、その内容は容易に想像がつきます。HTMLファイルが置いてあるディレクトリーの中のファイルに、CGIコマンドから取り込まれた情報を書き込んでいたのでしょう。どこからもリンクされていない、名前もわからないファイルは外部の者がアクセスできないと思っていたのかもしれません。しかし当然ながら、単純な名前のファイルであれば、その存在がすぐに明らかになってしまいます。独自のCGIコマンドではなく、雑誌などに掲載されたプログラムを真似しただけという場合、ファイル名は広く知られているはずです。

これらの例は技術的な側面もありますが、主に運用ポリシーの問題だと言えます。外部からアクセスされてはいけないものをアクセスできる範囲に入れておくというのが根本的な問題なのです。

扱っている情報のうち、何が公開してよい情報で、何が公開してはいけない情報なのかを明確に区別することが、まず求められます。おのおのの情報の管理ポリシーを明確にし、その管理ポリシーに従った管理を行わない限り、このような失敗は繰り返し発生するでしょう。

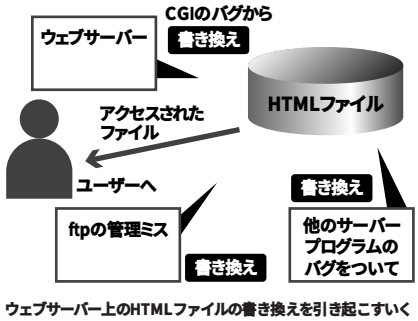
ずさんすぎる管理

ウェブサーバーの問題ではなくても、ほかのツールなどのセキュリティーホールの影響がウェブサーバー経由で出てしまうような例を挙げましょう。ある会社で、ウェブサーバーで公開している画像データが何者かに書き換えられるという事件がありました。その画像データはウェブサーバー外部から更新できるものだったのです。

このファイル更新にはFTPが使われましたが、その設定がなんとユーザー不特定パスワードなし（いわゆるanonymous ftp）だったのです。これでは外部からファイルを書き換えてくださいと言っているようなものです。この問題は先ほどの「運用ポリシーを明確に」という部分ともオーバーラップします。

セキュリティーのためのシステムをどんなに堅牢にしても、また、ソフトウェアのセキュリティーホールが皆無であっても、このようなずさんなシステム管理をしていては、まったく何の意味もなさなくなります。

図❶



ウェブサーバー上のHTMLファイルの書き換えを引き起こすいくつかの要因。

ウェブサーバーから防げない場合

自分の管理しているサイト側がどんなに完璧であっても、自分の守れる範囲以外の部分で問題が発生する場合があります。もっとも極端な例はドメイン名をIPアドレスに対応させるDNSのデータの偽造です。これによりドメイン名をIPアドレスに変換する際に誤ったものになってしまうため、目的のウェブサイトに正しくアクセスできないという問題が発生します。

1997年にはインターネットの中心DNSとも言えるInternicのDNSがこの攻撃を受けました。現在はさまざまな対策がとられてこのような問題は発生しないようにはなっていますが、同様の問題が再び発生しないとは言い切れません。この問題に関しては、自分のサイトではなく、DNSを管理している組織の問題なので、トラブルが生じてもじっと回復を待つしかありません。

このように自分では防ぐことができない問題も存在するのです。

ウェブページ改ざんの原因について

ウェブサーバー（Apache）自体に関して安全かと聞かれれば「かなり安全である」と答えられると思います。ではなぜ「ウェブサーバーに侵入されてHTMLファイルが書き換えられた」ということがよく聞かれるのでしょうか？　侵入されているのにどうして気がつかないのでしょうか。

答えは簡単です。HTMLを書き換える攻撃はウェブサーバーを経由しているのではなく、ほかのツールの脆弱性を突いているからです（図❶）。また「侵入」という言葉から、遠隔ログインでサーバーに侵入して作業を行っているように錯覚しますが、マシンにログインすることなしにHTMLを書き換えることもできます。たとえば先に例に挙げたように、FTPの設定がまったくセキュリティーの役に立っていないような場合、HTMLファイルを外部から自動的に書き込むことも可能です。

ウェブサーバーと同じマシン上で動くサーバープログラムにバッファーオーバーフローなどセキュリティーに関する重大なバグが存在していて、ウェブサーバー上で任意のコマンドを動かすことが可能になる場合は極めて深刻です。

このような場合、ウェブサーバー側から任意のコマンドを実行し、インターネット側からファイルをウェブサーバー上のHTMLファイルに上書きしてしまうといったこともできますし、最悪の場合、ウェブサーバー上のファイルがまるごと消去されてしまう危険性すらあります。これは外部から完全に自動化した形でアタックできるので、確率は低いとはいえ、これらのアタックは極めて深刻な事態を引き起こします。

信頼あるCGI以外は使わない

ウェブサーバー本体であるhttpdプログラムは問題ないとしても、ウェブサーバーと連動して外部から実行されるプログラムに欠陥が

DNS（ディー・エヌ・エス／Domain Name System）：インターネットで用いられるIPアドレスと名前を管理するしくみ。impress.co.jpなどといったインターネットで利用するドメインとマシンの名前は、すべてDNSによって管理されている。このDNSの運用を行うのがDNSサーバーで、クライアントのリクエストに応じて、ドメイン名やIPアドレスを返す。

あれば、そこからセキュリティーの問題が発生します。そのような意味でCGI（Common Gateway Interface）には常に問題が潜んでいます。CGIはcgi-binプログラムとも呼ばれ、ウェブサーバーから起動されるプログラムです。データはユーザーが使っているクライアント（ウェブブラウザー）からウェブサーバーに送られ、そしてCGIに渡されます。

このCGIの設計が悪いと、そこがセキュリティーホールになってしまうことがあります。利用しているhttpdの仕組み、CGIのメカニズム、そしてプログラミング言語の仕組みをよく理解したうえでCGIプログラムを作成することが重要です。

しかし、あちらこちらのウェブサイトで公開されている「CGIの作り方」を見てみると、安全な設計方法まで踏み込んでいる信頼性のある説明はわずかで、ほとんどは「～というのを作ってみました」というレベルで終わってしまっています。また雑誌での解説や参考本で解説しているCGIの作り方も似たりよったりです。

UNIXのシェルプログラムや、内部で直接コマンドを呼び出しているようなPerlプログラムでは、CGIの引き渡されたパラメーターがどう使われているかを注意深くチェックする必要があります。現在ではApacheが特殊文字はブロックしているとはいえ、セキュリティー上の致命的な問題を持っている可能性がありますので注意しなくてはなりません（図❷）。

インターネットのセキュリティー問題について報告をまとめている「CERT Advisory」 Jump でも、CGIに関しての危険性に注意を促しているドキュメントが複数あります。

リスト❶　access_log

```
http://localhostをアクセスしたログ

127.0.0.1 - - [04/Dec/1999:21:28:03 +0900] "GET / HTTP/1.0" 200 1982
```

CGIを使おうと思っている方は、これらのドキュメントに目を通しておく必要があるでしょう。もし、CGIプログラムが安全であるかどうか自分で確信が持てなければ、まずcgi-binにあるすべてのCGIコマンドの実行可能パーミッションを削除しておくべきです。

Jump ftp://ftp.jpcert.or.jp/pub/cert/cert_advisories

アクセスログを活用する

RedHatのRPMパッケージではウェブサーバーは2つのログを残します。/var/log/httpdにあるaccess_logとerror_logの2つです。前者（リスト❶）はhttpdへのアクセスのすべてを記録します。後者はエラーとなったアクセスのみを記録します。これらのログファイルは週単位でローテーションされており、また、1か月以上前のログは残らないので注意してください。

これらはマシンが不正アクセスを受けたり、エラーが発生したりしたときの状況を記した貴重な記録になりますので、有効に活用してください。

外からのチョッカイ

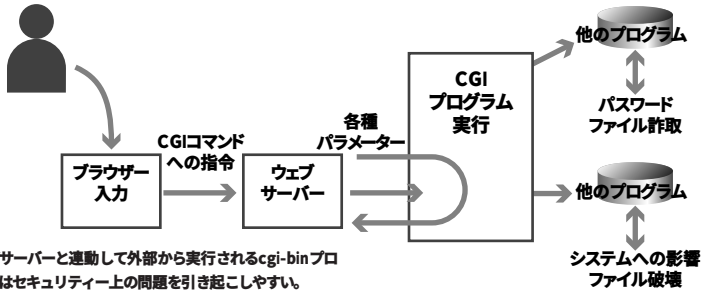
ずいぶん以前のhttpdのcgi-binには、デフォルトでhandler、phf、test-cgiなどのコマンドが用意されており、これらのコマンドにはセキュリティーホールがあったため、ここを突いて数々の攻撃が可能でした（ちなみに、本連載の対象としているパッケージであるRedHat 5.2に収録されたApacheパッケージにはこれらのコマンドは含まれていませんので心配はありません）。

いまでもこれらのcgi-binコマンドに対してチョッカイを出してくる連中がいます。外部から接続可能なウェブサーバーがあるとアタックを試みるための事前の探知（probe）が行われます。これはそのマシン上でhttpdが動作しているかどうか、あるいは外部に公開しているかどうかなどお構いなしです。まず、すべてのIPアドレスをアクセスしていきます。次に、そのマシンからのウェブサーバーの反応があれば（ポート80へのリクエストの反応があったら）、次に既存のセキュリティーホールがあるかどうかのチェックを始めます。

最初は無難にセキュリティーホールのあるcgi-binコマンドが存在しているかどうかだけをチェックします。これだけなら外部に公開しているcgi-binを実行したにすぎませんから探知している側の責任は問われません。この後、これらの情報をもとにそのウェブサーバーにアタックをかける段階へとエスカレートしていきます。もっと悪意のあるものは、確認の手順を踏まずに直接攻撃してきます。

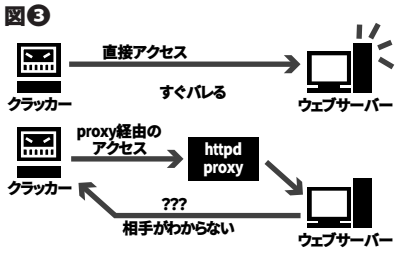
ウェブサーバーの運用を始めれば、すぐにこのようなcgi-binコマンドを探知した形跡や攻撃のログを見つけることになります。しかし、慌てることはありません。なぜなら、ここまでのアドバイスをきちんと守っているならcgi-binの中には上で記したようなセキュリテ

図❷



ウェブサーバーと連動して外部から実行されるcgi-binプログラムはセキュリティー上の問題を引き起こしやすい。

図❸



内部のネットワーク情報を外部に公開しないためのセキュリティーツールであるhttpd proxyが、不正アクセスの足跡を消す便利なツールとして使われてしまっている。

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。

ィーホールを持つコマンドは一切入っていないはずだからです。

どこから攻撃が来るのか

cgi-binを経由して攻撃をしかけてくるものは、直接サーバーにアクセスすると確実にIPアドレスなどの足跡を残します。ですから、ほとんどと言っていいほどhttpd proxyを経由し、足跡を消してアクセスしてきます。

httpd proxyはサイト内部に向けてHTMLをキャッシュしてネットワーク効率を上げたり、内部情報を外部に公開しないために使われたりしています。現在では、個人ユーザーであってもキャッシュによる効率化を考慮して利用している場合もあります。

このhttpd proxyを経由したcgi-binへのアタックは十分に踏台攻撃の踏台として利用できます。よく「ホストを乗っ取られて、他のホストを攻撃する踏台に使われる」という説明があります。cgi-binを使ってのシステムへの攻撃は（通常の動作範囲である）proxyで十分なのです。皮肉にも内部のネットワーク情報を外部に公開しないためのセキュリティーのツールとして機能する部分が、不正アクセスの足跡を消す便利なツールとして使われてしまっているのです（図❸）。

このhttpd proxyは意図的に外部にオープンしているものか、アクセスコントロールを間違えているのか、あるいは管理がずさんでアクセスコントロールされていないのかはわかりませんが、とにかくかなりの数のhttpd proxyがオープンなままであるようです。一方で外部からも利用できるhttpd proxyに関しての問題は一般に広くは取り上げられていないようでもあります。

SOHOサイトを自分で管理してみるとわかるのですが、インターネット側のルーターのリジェクトログ（*1）を見れば、国内、海外を問わず、httpd proxyに使われているポート番号をねらってのアクセスの多さに驚くはずです。httpd proxyにアクセスすること自体は不正アクセスともセキュリティーホールとも呼べませんが、外部からの利用を許したhttpd proxyは不正アクセスの踏台を提供しているのと同じことになります。インターネット側のルーターではhttpd proxyへのアクセスを許してはいけません。

（*1）通過を許可しないIP接続の接続拒否記録。後に行うルーターの回で説明する予定です。

MOなどを使ったローテクなプロテクション

ウェブサーバー上のHTMLファイルの書き換えは、あちらこちらの有名なサイトで（CIAのウェブサーバーですら）攻撃されている、現実的、かつ広く認識されている問題でしょう。

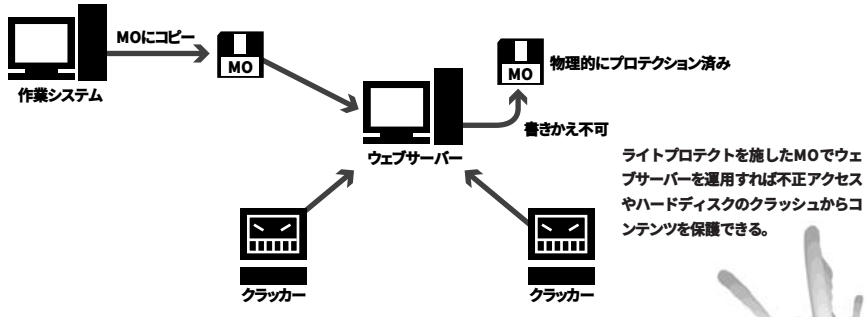
これは「ファイルの書き換え」ということが不可能であれば発生しません。そのような状態をhttpdを動かすマシンで実現してしまえばいいのです。一番簡単な方法がMOを使う方法です。まず内部ネット上にあるLinuxマシン上でhttpdを動かし、HTMLの記述と動作チェックを行います。次にその作業用のLinuxマシンのディレクトリー/home/httpd以下をまるごと、ファイルシステムを構築したMOにコピーします。MOを外部に公開するサーバーに持っていき、セットします。MOのライトプロテクションを「書き込み不可」に設定し、MOのデバイスをを/home/httpdにマウントします。その後、httpdをスタートさせます。この方法だとHTMLのメンテナンスは楽ですし、ファイルを外部から書き換えるといった攻撃には、ほぼ万全です（図❹）。

もう1つ良い点は、実はMOで動かしているためにウェブサーバー上のデータは別のマシンにバックアップされていることになります（本当はウェブサーバー側がバックアップで動いている）。ウェブサーバーのディスクがクラッシュして情報をすべて失ったという話をよく聞きますが、この方法だとウェブサーバーのクラッシュはコンテンツに対しては影響与えません。万が一悪意のある侵入者がウェブサーバーに侵入できて、そのディスクの内容を完全に破壊したとしてもコンテンツは無事です。

唯一の弱点はMOの読み込み速度がハードディスクより遅い点です。しかし、SOHOレベルでの通信のボトルネックは常にネットワーク回線の速度ですので、大きな欠点とまではいかないはずです。Linuxのファイルシステムは搭載メモリー量に合わせてキャッシュ効率が上がるので、よく利用されるファイルは（2回目以降のアクセスに限りますが）ハードディスクもMOもあまり変わらない速度でアクセスできます。

以上はポピュラーなMOの例ですが、容量が100Mバイト以下で十分であればZipでも同じことができます。また逆にギガバイト単位であればDVD-RAMという選択肢もあります。SOHOサーバーの場合、ISPや企業のように容量の大きいシステムが不必要で、管理者とコンテンツデザイナーが近くに位置し（あるいは同一人物で）、ハードウェアに直接アクセスできるので、このように小回りが効く方法が考えられるのです。SOHOはSOHOなりに、大きな企業ではできないような利点もあると言えるでしょう。

図❹　サイトのネットワーク構成図



proxy（プロキシー）：社内ネットワークなどからインターネットに接続するとき、内部のセキュリティーを守るために境界に置く代理サーバーのこと。
また、WWWアクセスにおいては、HTMLデータなどをキャッシュし、同様のリクエストに対して蓄積したデータを返すことによって、クライアントのWWWアクセスを高速化できる。

［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