

# Mask Your Web Server for Enhanced Security

By Joe Lima

Masking or anonymizing a Web server involves removing identifying details that intruders could use to detect your OS and Web server vendor and version. This information, while providing little or no utility to legitimate users, is often the starting place for crackers, blackhat hackers and "script kiddies". This article explores some ways you can minimize the risk of such detection. Most of the following examples focus on Microsoft's Internet Information Services (IIS) Web server, since it has been most widely lambasted for its vulnerabilities, but some Apache detection countermeasures are also covered. While IIS users probably have the most vested interest here, server anonymization is relevant to anyone responsible for administering a Web server.

Crackers Start Here. Shouldn't You?

Let's look at it from the attacker's point of view: Security vulnerabilities tend to be dependent on software vendor and version. Blind probing might lead to further requests being denied or a system temporarily taken off line. Knowing Web server details greatly increases the efficiency of any attack. If an attacker can target exploits, the chances of successful cracking prior to detection increase significantly. Script kiddies can leverage canned, newly-discovered exploits to do more damage faster by targeting hosts with recognizable signatures. A self-identifying system invites trouble.

Port80 Software has developed an IIS server module called ServerMask to combat the majority of issues explored here for the Windows Web Server.

## The Server Header Tells All

Most Web servers politely identify themselves and the OS to anyone who asks. Using a network query tool like Sam Spade or this Header Check, you can discern the HTTP Server header. Just request a Web site's home page and examine the resulting HTTP headers or "banners" sent back by the server. Among them, you will likely find something like this:

```
Server: Microsoft-IIS/5.0
```

There is not much mystery here. Apache's default settings make it no less identifiable:

```
Server: Apache/2.0.41-dev (UNIX)
```

You can remove or obscure this HTTP Server header in a variety of ways, depending on your platform. Apache 2.x users who have the mod\_headers module loaded can use a simple directive in their httpd.conf file, as follows:

```
Header set Server "New Server Name Goes Here"
```

Unfortunately, mod\_headers cannot alter the Server header in prior versions of Apache. IIS users can install IISLockDown and use the configuration option in URLScan's INI file for removing or replacing the header. Be careful with URLScan if you are using Cold Fusion application server -- the way the current version replaces the Server header wreaks havoc with CFM pages. In fact, removing the header is the way to go when using URLScan, since if you try replacing the header it moves to the bottom of the header order -- which pretty much gives away that you are running URLScan on IIS.

## Unseen File Extensions

Displaying file extensions like .asp or .aspx in a site is a clear indication that you are running a Microsoft server and, in general, hiding file extensions is a good practice to mask the technology generating dynamic pages. You can change your application mappings (.asp becomes .htm or .foo, etc.), but such one-to-one mapping can make mixing server-side technologies painful and does nothing to alleviate headaches during site migrations. Doing without file extensions altogether is an even better idea, not only for security but also for ease-of-migration and content negotiation. Apache people will want to take a look at mod\_negotiation. Watch out, though, for the Content-Location header in the server's response, which can give away the file extension that is not shown in the URL. You might have to suppress this header separately using mod\_headers. In a similar vein, Port80 offers a tool called PageXchanger that allows file extension hiding in IIS.

## Half-Baked Cookies

The ASP session ID cookie, used by the Session object to maintain client state, is another dead giveaway:

Set-Cookie: ASPSESSIONIDQGQGGWFC=MGMLNKMDENPEOPIJHPOPEPPB; You can disable ASP Session State so that this cookie is not placed, but you lose the convenience of using the Session object to maintain client state. You could also create an ISAPI filter to change the names of any session ID cookie. On the other hand, ASP sessions are resource intensive, and turning them off improves the performance and scalability of your ASP application, while also

helping to anonymize your server.

#### Send These to the Recycle Bin

WebDAV: Another way of identifying Microsoft servers is their implementation (from Windows 2000 and IIS 5.0 on) of WebDAV -- the HTTP Extensions for Distributed Authoring and Versioning. WebDAV itself is not unique to Microsoft or IIS; it is a proposed standard (RFC 2518) with an IETF Working Group. Microsoft's WebDAV support, however, adds a lot of information to the headers sent back by the server, especially when an HTTP OPTIONS request is made. If you are not using WebDAV (to support Outlook Web Access or Web Folders, etc.), you can disable it entirely by editing the registry or by using IISLockDown and URLScan.

Other Headers: Certain Web servers betray their identity by displaying specific headers in HTTP responses, and four popular IIS headers should be removed or at least examined. The X-Powered-By and X-AspNet-Version headers are obvious signs that you are running ASP.NET and therefore some flavor of IIS. Few popular Web servers send the Public header in response to OPTIONS requests (while almost all respond with the similar Allow header). The presence of Public is a good indication you are connected to either an IIS box or Netscape Enterprise 3.6 and should be masked. Last but not least, mask the MicrosoftOfficeWebServer header -- while this will not effect FrontPage publishing, you should test this header obfuscation if you are using these server extensions for any other purpose.

Integrated Windows Authentication: IIS users should not rely on "Integrated Windows Authentication" -- especially not as a way of hiding anything on the server. This method betrays the very secret it would keep, since a script or visual hacker can identify the Windows box by means of the WWW-Authenticate headers sent by the server. When a file or directory is protected by NT Challenge-Response authentication, one of the authentication headers contains the string "NTLM" (NT LAN Manager) -- a Microsoft-specific form of HTTP authentication.

#### Get Your Headers Straight

The number and sequence of your HTTP headers and the presence or absence of certain platform-specific headers provide handy ways for more sophisticated hackers to fingerprint your Web server. A relatively unexplored area of server profiling, this will become a more common exploit as administrators start to implement countermeasures against obvious HTTP vulnerabilities like the Server header. For IIS users, a custom ISAPI filter can alter the Microsoft-specific header order or sequence to emulate, say, a default Apache installation. Apache users can accomplish any header order emulation they wish by experimenting with the location and order of Header directives in mod\_headers.

#### Whose Default is That?

Default messages, pages and scripts of all kinds often contain clues to server identity, and these should be removed or modified accordingly. Software behind the Web server often bubbles error messages back through the HTTP request/response cycle, and customized HTTP errors can mask application server, database server, Web server and OS identity. For IIS, CustomError makes it easy for developers to deploy custom 404 and other HTTP error pages. This article shows how to implement custom HTTP errors in Apache. Avoid this on a development server, since, when done properly, it prevents database and server-side scripting errors from being seen -- making it tough for developers to debug their applications! Remove or hide any Web or application server administration pages, scripts or documentation installed under your server's Web root, and make sure to replace those default home pages.

#### We Don't Need No Extra Services

Beyond the HTTP service itself, many computers used as Web servers host a number of other network services. Perhaps the most common are FTP and SMTP. As a general security rule, try to avoid running these services on your Web server. In particular, avoid the default FTP and SMTP services in Microsoft IIS. Despite the convenience of integrated services, there is no reason to have Web, FTP and SMTP services interlinked. This is not an issue for Apache, since the Web server is not associated with FTP and SMTP services through a common administrative service. If you do use these services, be aware that they will advertise your IIS server's identity.

When a connection is established with an SMTP service, the recipient server sends a human-readable greeting to the client, the "SMTP banner". What the SMTP banner displays has no effect on e-mail service, but, like the HTTP Server header, it divulges details about the software running on the box. The default Windows SMTP service exposes such information. To find out how to change the SMTP banner, check here.

The default Microsoft IIS FTP server also presents a known banner. Since modifying the FTP banner is a more involved process than modifying the SMTP banner (plan on hacking several system DLLs), your best bet is an alternate FTP server like RhinoSoft's Serv-U FTP server that can display any text message in the FTP banner. As an added bonus, third-party FTP servers like Serv-U are more configurable than the IIS FTP service when it comes to security measures like assigning users their own login directories.

#### Unsanitary Inputs

Many platform-specific exploits use complex URL strings to gain access to a shell or CGI program, from which a hacker can easily get a directory listing revealing the OS' default file structure. Once a shell or CGI program is hijacked and the file system revealed, the door is wide open. The best defense against this trial-and-error exploit is a user-input filter or

"sanitizer" that removes unacceptable characters (such as meta-characters and their various possible encodings) from user-supplied data. For IIS, the current standard is IISLockDown/URLScan. A new generation of application firewalls extend this protection to the application layer behind the Web server. In the Apache world, user input sanitizing is traditionally the responsibility of CGI authors. Here is the classic CERT article on the topic, with examples in Perl and C. If you are setting up a new box, consider changing the default file structure as well. Input sanitizing and rearranged file structures do double duty -- helping to disguise the box and neutralize common exploits simultaneously.

#### Combing Through the Stacks

Even when all telltale signs are removed from your Web server's application layer, there remain detection weaknesses at lower network layers. Any server with a network connection has a network protocol stack subject to being scanned and identified. The best stack scanners (like NMAP from insecure.org) can ID most operating systems by using a variety of techniques to fingerprint the system's TCP stack. OS-specific IP stacks are also vulnerable to detection via the Internet Control Message Protocol (ICMP), used by the popular Ping utility. Good resources on ICMP vulnerabilities can be found here. The first line of defense against these kinds of network scanning vulnerabilities is a good, well-administered firewall. However, careful network analysis can still identify a box by examining the packets a firewall must permit a Web server to pass through in response to HTTP requests.

#### Netcraft is Watching

Take a look at the "What's that site running?" tool on Netcraft. If you point the site profiling tool at your own Web site, it will probably correctly report both your Web server and OS. Changing your HTTP Server header will cause Netcraft to report a false value for your Web server -- or just "unknown" if the header is completely removed (the change is not immediate, as Netcraft caches results for a time).

Still, your OS will probably be correctly identified -- even behind a good firewall. To get Netcraft to report your OS as "unknown", you will have to tinker with some of your default TCP/IP settings, such as the receive window size (RWIN), the maximum transmission units (MTU), the maximum segment size (MSS), and/or the IP header time-to-live (TTL). Altering these settings will affect your server's performance in diverse ways, depending on network conditions, so considerable care should be taken when changing these defaults. In the hands of a skilled network administrator, however, this strategy can be an effective countermeasure to information leakage through stack scanning.

#### Let's Be Careful Out There

No combination of detection avoidance succeeds in completely anonymizing your Web server -- just as no combination of firewalls, IDS, and other security countermeasures can completely defeat a skilled and determined cracker. Like server hardening, server anonymization can help defeat the majority of would-be attackers. And like all aspects of network security, it's a never-ending battle to stay ahead of the bad guys.

#### About the Author

Joe Lima is the Director of Product Development for Port80 Software. He has worked for a variety of Internet, wireless and software development companies, specializing in research and development for server-centric technologies. Visit [port80software.com](http://port80software.com).