An ICG Technology White Paper

Web Server Security Hardening Best Practices, Version 01

Adopting hosting default or legacy configurations can leave your vulnerable – don't get caught exposed.

July 10, 2017

IowaComputer GURUS

Contents

The Purpose of This Document

This document has been created to provide a baseline set of procedures and recommendations for securely configuring web application servers and services. This document is not intended to be an introductory course on website security. Neither is it a comprehensive set of website or server security guidelines. Rather, it is designed to aid in the application and implementation of security configurations – some of which are commonly overlooked – that could leave a website vulnerable.

Who Should Use This Document

This documentation is intended to be used by individuals with the technical skills and systems permissions to access and understand the hosting environments used by their web properties. It may also be of value to website administrators using third-party hosting solutions to start a conversation on server security with their IT infrastructure resource providers.

Readers should exercise caution when implementing the recommendations in this guide, as incorrect implementations could result in additional security holes or system downtime if not implemented correctly.

Document Revision History

Rev. Date	Notes
7/10/2017	First public release, Version 01.

Web Server Security Hardening - Background

Website security has become the fundamental concern of the internet. According to recent statistics, there are well-over 1 billion websites,[1] most of them relying on out-of-date platforms and utilizing lax or outdated security.

The number and severity of security threats is growing every year. Computer security market leaders McAfee[2] and Symantec[3] have both recently issued threat assessment reports that suggest that all forms of internet security are under persistent and growing levels of attack. Further, in many instances these attacks can be defended against through the vigilant application of best practices and coordination.

Often, when reviewing historical information regarding exploits it is uncovered that the malicious actions were taken by exploiting an exposed or existing vulnerability in the platform. Some recent attacks have leveraged security holes that allow unauthorized users to upload files. There are many online resources discussing software-based defenses to this kind of an attack. However, these articles can fail to discuss the various methods to limit exposure via proper server configurations.

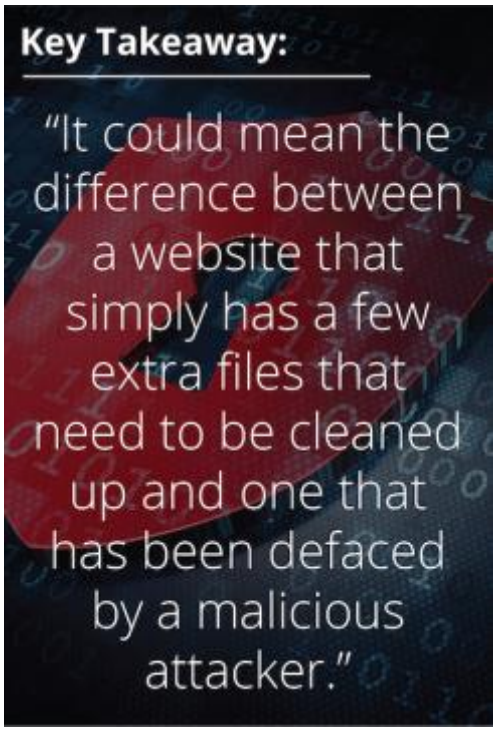
The Open Web Application Security Project (OWASP) lists security misconfiguration in its top 10 threat listing, and has included it in not only the 2010 & 2013 finalized versions, but also in the updated 2017 draft version.[4]

In an effort to help administrators understand common misconfigurations in their environment we have prepared this best practice guide to fill this information gap.

[1] Most recent Netcraft News report as of this writing, May 26, 2017.
[2] McAfee Labs 2017 Threats Predictions: https://www.mcafee.com/us/resources/reports/rp-threats-predictions-2017.pdf
[3] Symantec Internet Security Threat Report 2017: https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf
[4] The Open Web Application Security Project Top 10: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

Disable Unused Technologies (for .NET Applications)

Many hosting environments provide a robust collection of supported technologies, and it is common for installations to be created in a manner that enables more frameworks in the environment than necessary. As a leading provider of ASP.NET technology solutions, we are very familiar with the common pitfalls for those working with ASP.NET. However, similar processes and practices could be directly applicable to other technologies and development platforms.



The risk of exposure from supporting unused technology includes the potential for a user to upload or place executable code on your website in a different programming language. In some automated, script-based attacks, this simple process could be the difference between a website that simply has a few extra files that need to be cleaned up and one that has been defaced by a malicious attacker.

For example, Cold Fusion is a technology supported by many hosting environments. But Cold Fusion is rarely used in modern website creation. Leaving Cold Fusion enabled on your server can provide an opportunity for a hacker to place Cold Fusion-based malicious code on your server where it would remain invisible, but executable.

By ensuring that your server is configured to only support the platforms and technologies necessary to support your particular site and applications removes this risk.

Disable Classic ASP

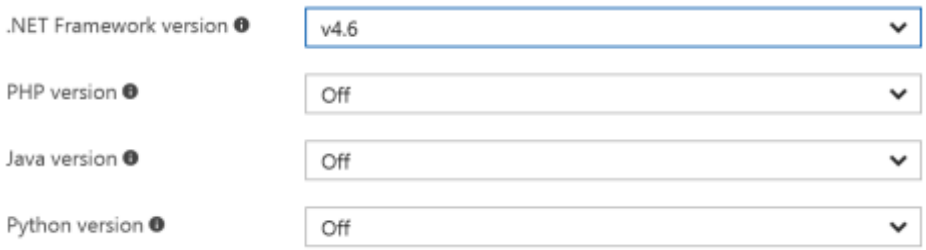
When reviewing installations and server configurations, we have found that the most often enabled unnecessary coding language is Classic ASP. Classic ASP – or simply ASP – refers to the technology that preceded the current ASP.NET technology. Code written in Classic ASP is written in Visual Basic 6, and is typically applicable to legacy applications.

Yet default server configurations of Windows Servers will end up including support for Classic ASP, even if the installing/configuring party did not intend to do so. This can be disabled at the server level or at the individual website level within your IIS Configuration. This is accomplished by simply removing the HTTP Handler mapping within IIS for "ASP Classic." This will prevent any .asp files from executing and eliminate this risk.

Disable PHP, Java, Python or others

For years, those deploying applications using control panels (e.g.,Plesk, CPanel) as well as newer solutions using public cloud offerings (such as Azure App Services) could choose their technology package on an application-by-application basis. However, developers are more focused on ensuring their desired technologies are configured in an acceptable manner – often overlooking any other installed configuration.

The methods to disable PHP, Java, and Python support will vary depending on the hosting provider and solution used by your application. The following is a sample taken from a Windows Azure App Service account.



In this example, you can see that we have turned off support for PHP, Java and Python. The default configuration of this application – prior to changes – did include PHP & .NET, however Java and Python were disabled.

Manage & Limit Default Documents

When a user requests a URL that is a path to a folder on a web server, the default document setting is used to determine what code is served. When a user requests your domain, such as www.mysite.com, it will look in the root folder of the application for files matching the file name in the root directory. The screenshot shows a default documents configuration from a Windows Azure App Service, IIS configuration.

As you can see from this sample listing, a number of files are included. However, if we are working with an ASP.NET application we only need default.aspx as our default document type.

This list is processed in order by the web server based on the configuration. If a user was able to load a file to the folder with a name that appears earlier in the listing than default.aspx (e.g., index.html, iisstart.htm, etc.), it would be possible to override the homepage document.

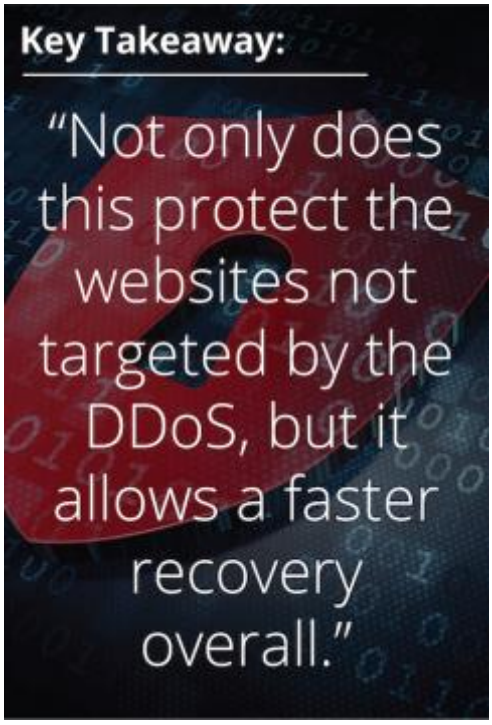
Removing all items, except for Default.aspx is typically recommended for all ASP.NET based solution to limit the ability for a malicious user to take control of your website assets.

Managing Application Pools

Simple searches on Google for "Application Pool Best Practices" will display an almost religious debate between developers regarding the best process to follow. We don't want to re-hash this debate. However, we want to emphasize a few baseline, generally-accepted considerations to ensure that proper decisions have been made with regard to your server environment.

Be Cautious of Multiple Apps Per Pool

A "one application pool per application process" is the default. This configuration will use more server resources than other configurations, so many times developers or administrators looking to reduce application overhead will change server settings to share an application pool across multiple applications.



It is true that allowing applications to share pool resources will reduce system overhead. But it is important to understand that this would also mean that if ANY application in the pool is exploited or otherwise compromised – even in a non-malicious manner – it could impact the other sites within the same pool.

A classic example of this if a single website experiences an Distributed Denial of Service (DDoS) attack. If the target website is sharing an application pool with other websites, the attack could take impact all websites and applications in the pool. If an application boundary is identified with a pool, any attack of this nature can usually be isolated to that website. Not only does this protect the websites not targeted by the DDoS, but it allows a faster recovery overall.

Additionally, when two applications share the same pool, they are working under the same user account. This makes it possible for an exploited application to manipulate files in other applications because they share security context.

It is for these reasons that we recommend following the default behavior of one pool per application.

Sharing Service Accounts Across Pools

Years ago, if you followed installation guides for a new application within IIS you would find that the "easy" solution was to use "NETWORK SERVICE" as the user account. This process appears to be secure. However, by sharing a service account across multiple applications you open a door for malicious users to attack more than one web application should it be able to gain unauthorized access to the file system.

This practice essentially negates one of the key benefits of having multiple application pools by allowing services to potentially cross application boundaries with file access and other

manipulations. You should always ensure that each application has its own identity for security and logging purposes.

Managing Server Updates

Security vulnerabilities can come in many forms. Some attack individual applications, while others target the server. Just as it is important to ensure that the most current and secure application versions are deployed, it is also important to ensure that all servers are patched properly. This includes ensuring that Microsoft Windows, SQL Server, the .NET Framework, and any other server side technology is updated on a disciplined, regular schedule.

Don't Forget Your Control Panel

In our experience assisting customers with identifying security risks, we have found that – even when administrators have a defined process in place to update Windows, SQL Server, and their Anti-virus platforms – they often overlook the administration control panel (e.g., Plesk, cPanel, etc.). Control panels are also susceptible to security risks and vulnerabilities, so be sure to include regular updates to these management systems.

Application Firewall Configuration

As cloud-based solutions are becoming more prevalent, the ease of implementation and lower costs make application firewalls even easier to install. For example, Windows Azure SQL Databases include firewall rules that prohibit access from outside of the application by default.

For security, you will want to limit the number of open ports, services, and applications as much as possible. By limiting the number of openings, you can limit the amount of traffic. The following are a few key recommendations in this area.

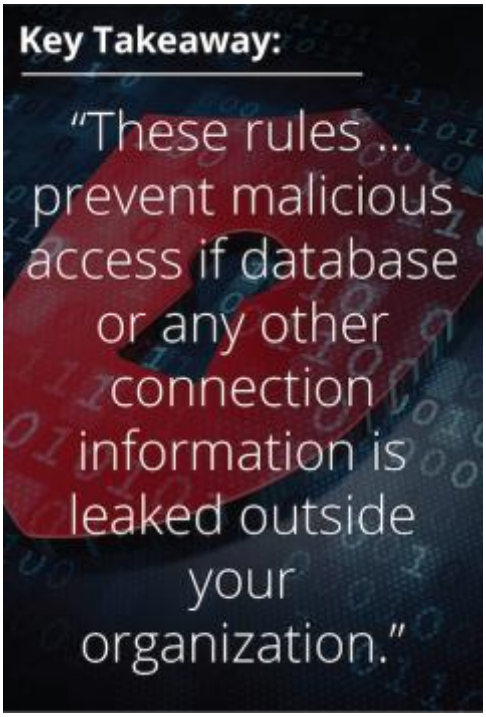
Support SFTP vs FTP

When selecting a deployment technology, the use of SFTP adds an extra layer of security during the transmission of files to and from your server. With this in mind, if you have access to support SFTP you might consider to disable – or strongly restrict – access via a straight FTP protocol.

Restrict Elevated Access by IP or VPN

Certain ports, such as those necessary to serve your application, must be left open. However, additional services such as Remote Desktop Management, SFTP, or Remote SQL Server access can and should be restricted to a subset of IP addresses or secured via a VPN solution.

In our installations in Azure we use application firewall rules to restrict database access to developers on a need to know basis, including for development environments. These rules not only protect you in the case of a random attempted attack, but also can prevent malicious access should database or other connection information be leaked outside of your organization. The loss of a configuration file with a database user password is a big deal. However, the risk ls limited should the information not be executable or otherwise of value to the receiving party.



Monitor Server Activity

Application servers are most commonly "hands off" environments that are managed only during a crisis event. However, it is important to understand the benefits of server activity reporting for security purposes.

By using commercially available server monitoring tools you can get a recording of the server activity, including incoming requests, errors, and other important information. Spikes in activity or activity at odd hours of the day can indicate that immediate attention is required, even if the application itself seems to be unharmed.

Monitoring solutions have recently become more affordable. We strongly recommend researching and implementing a solution as soon as possible if you don't have one already. Available solutions include, but are not limited to:

- CopperEgg
- NewRelic
- Application Insights

Store Server Logs

If an exploit does occur, it can often be beneficial to "work backwards" to diagnose what happened. This is only possible if application server logs have been retained. The log files can take up considerable disk space for active websites, but this downside can be successfully managed via rules-based scripting that ensures things don't get out of hand.

In the event of an attack or exploit, server logs can provide invaluable information that identifies what was done, who did it, and when exactly it was done. It is our recommendation to retain at least 30 days of log information to ensure the best ability to research any potential issues.

Summary

While the security best practices provided in this document are not all-inclusive, they form the core of what we consider to be relatively-easy to implement website server security policies that protect against most common attacks. Perhaps surprisingly, most active websites do not employ even the most basic of these techniques.

Website security will continue to be a moving target as technology evolves and malicious attacks continue to evolve with it. IowaComputerGurus is committed to remaining at the forefront of complete website security to protect our customers and the larger internet community.

If you want to ensure that you are viewing the latest version of this document or want advice and recommendations specific to your environment and use case, please let us know. We are always happy to help.

About IowaComputerGurus Inc.

IowaComputerGurus Inc., a Microsoft Certified Partner organization specializes in developing custom solutions using the Microsoft .NET development stack and leading Content Management Systems. Based in Des Moines, Iowa, they provide services to customers all over the world and base their business on providing quality, affordable technology solutions with the best customer service. The company is led by Mitchel Sellers, a Microsoft MVP, AP.Net Insider, Microsoft Certified Professional, DNN MVP, and published technology author.

Contacting IowaComputerGurus

IowaComputerGurus, Inc.
5550 Wild Rose Lane, Suite 400
West Des Moines, Iowa 50266

Phone: (515) 270-7063
Fax: (515) 866-591-3679
Email: webmaster@iowacomputergurus.com
Website: http://www.iowacomputergurus.com

Feedback

IowaComputerGurus is committed to quality and appreciates constructive feedback on our Best Practices documents, white papers, and other technical guides. If you discover any errors or have suggestions of additional items for inclusion or any modifications to existing content, please use one of the above listed contact methods to let us know.

Disclaimer

This document reflects the opinions and experience of the authors as a non-compensated service to the community. It is provided "as-is," and no warrantee is expressed or implied as to the serviceability or applicability of any information or recommendation for any particular purpose, application, and / or environment. It is the reader's responsibility to conduct their own diligent research, consult experts, and determine whether this information is right for their website, application, and / or environment. Additionally, the reader understands that the use of this documentation constitutes agreement to the any additional relevant terms of use as currently published on the IowaComputerGurus.com website, which are subject to change.

Copyright and Trademark Notices