Security Benchmarks™
A DIVISION OF CENTER FOR INTERNET SECURITY

CIS Microsoft IIS 8 Benchmark

v1.0.0 - 01-06-2014

The CIS Security Benchmarks division provides consensus-oriented information security products, services, tools, metrics, suggestions, and recommendations (the "SB Products") as a public service to Internet users worldwide. Downloading or using SB Products in any way signifies and confirms your acceptance of and your binding agreement to these CIS Security Benchmarks Terms of Use.

CIS SECURITY BENCHMARKS TERMS OF USE

BOTH CIS SECURITY BENCHMARKS DIVISION MEMBERS AND NON-MEMBERS MAY:

- Download, install, and use each of the SB Products on a single computer, and/or
- Print one or more copies of any SB Product that is in a .txt, .pdf, .doc, .mcw, or .rtf format, but only if each such copy is printed in its entirety and is kept intact, including without limitation the text of these CIS Security Benchmarks Terms of Use.

UNDER THE FOLLOWING TERMS AND CONDITIONS:

- **SB Products Provided As Is**. CIS is providing the SB Products "as is" and "as available" without: (1) any representations, warranties, or covenants of any kind whatsoever (including the absence of any warranty regarding: (a) the effect or lack of effect of any SB Product on the operation or the security of any network, system, software, hardware, or any component of any of them, and (b) the accuracy, utility, reliability, timeliness, or completeness of any SB Product); or (2) the responsibility to make or notify you of any corrections, updates, upgrades, or fixes.
- **Intellectual Property and Rights Reserved**. You are not acquiring any title or ownership rights in or to any SB Product, and full title and all ownership rights to the SB Products remain the exclusive property of CIS. All rights to the SB Products not expressly granted in these Terms of Use are hereby reserved.
- **Restrictions**. You acknowledge and agree that you may not: (1) decompile, dis-assemble, alter, reverse engineer, or otherwise attempt to derive the source code for any software SB Product that is not already in the form of source code; (2) distribute, redistribute, sell, rent, lease, sublicense or otherwise transfer or exploit any rights to any SB Product in any way or for any purpose; (3) post any SB Product on any website, bulletin board, ftp server, newsgroup, or other similar mechanism or device; (4) remove from or alter these CIS Security Benchmarks Terms of Use on any SB Product; (5) remove or alter any proprietary notices on any SB Product; (6) use any SB Product or any component of an SB Product with any derivative works based directly on an SB Product or any component of an SB Product; (7) use any SB Product or any component of an SB Product with other products or applications that are directly and specifically dependent on such SB Product or any component for any part of their functionality; (8) represent or claim a particular level of compliance or consistency with any SB Product; or (9) facilitate or otherwise aid other individuals or entities in violating these CIS Security Benchmarks Terms of Use.
- **Your Responsibility to Evaluate Risks**. You acknowledge and agree that: (1) no network, system, device, hardware, software, or component can be made fully secure; (2) you have the sole responsibility to evaluate the risks and benefits of the SB Products to your particular circumstances and requirements; and (3) CIS is not assuming any of the liabilities associated with your use of any or all of the SB Products.
- **CIS Liability**. You acknowledge and agree that neither CIS nor any of its employees, officers, directors, agents or other service providers has or will have any liability to you whatsoever (whether based in contract, tort, strict liability or otherwise) for any direct, indirect, incidental, consequential, or special damages that arise out of or are connected in any way with your use of any SB Product.
- **Indemnification**. You agree to indemnify, defend, and hold CIS and all of CIS's employees, officers, directors, agents and other service providers harmless from and against any liabilities, costs and expenses incurred by any of them in connection with your violation of these CIS Security Benchmarks Terms of Use.
- **Jurisdiction**. You acknowledge and agree that: (1) these CIS Security Benchmarks Terms of Use will be governed by and construed in accordance with the laws of the State of Maryland; (2) any action at law or in equity arising out of or relating to these CIS Security Benchmarks Terms of Use shall be filed only in the courts located in the State of Maryland; and (3) you hereby consent and submit to the personal jurisdiction of such courts for the purposes of litigating any such action.
- **U.S. Export Control and Sanctions laws**. Regarding your use of the SB Products with any non-U.S. entity or country, you acknowledge that it is your responsibility to understand and abide by all U.S. sanctions and export control laws as set from time to time by the U.S. Bureau of Industry and Security (BIS) and the U.S. Office of Foreign Assets Control (OFAC).

SPECIAL RULES FOR CIS MEMBER ORGANIZATIONS: CIS reserves the right to create special rules for: (1) CIS Members; and (2) Non-Member organizations and individuals with which CIS has a written contractual relationship. CIS hereby grants to each CIS Member Organization in good standing the right to distribute the SB Products within such Member's own organization, whether by manual or electronic means. Each such Member Organization acknowledges and agrees that the foregoing grants in this paragraph are subject to the terms of such Member's membership arrangement with CIS and may, therefore, be modified or terminated by CIS at any time.

Table of Contents

Overview

This document, CIS Microsoft IIS 8 Benchmark v1.0.0, provides prescriptive guidance for establishing a secure configuration posture for Microsoft IIS 8. This guide was tested against Microsoft IIS 8 running on Microsoft Windows 2012. To obtain the latest version of this guide, please visit http://benchmarks.cisecurity.org. If you have questions, comments, or have identified ways to improve this guide, please write us at feedback@cisecurity.org.

Intended Audience

This document is intended for system and application administrators, security specialists, auditors, help desk, and platform deployment personnel who plan to develop, deploy, assess, or secure solutions that incorporate Microsoft IIS 8.

Consensus Guidance

This benchmark was created using a consensus review process comprised subject matter experts. Consensus participants provide perspective from a diverse set of backgrounds including consulting, software development, audit and compliance, security research, operations, government, and legal.

Each CIS benchmark undergoes two phases of consensus review. The first phase occurs during initial benchmark development. During this phase, subject matter experts convene to discuss, create, and test working drafts of the benchmark. This discussion occurs until consensus has been reached on benchmark recommendations. The second phase begins after the benchmark has been published. During this phase, all feedback provided by the Internet community is reviewed by the consensus team for incorporation in the benchmark. If you are interested in participating in the consensus process, please visit https://community.cisecurity.org.

Typographical Conventions

The following typographical conventions are used throughout this guide:

Convention	Meaning
`Stylized Monospace font`	Used for blocks of code, command, and script examples. Text should be interpreted exactly as presented.
`Monospace font`	Used for inline code, commands, or examples. Text should be interpreted exactly as presented.
<italic font in brackets>	Italic texts set in angle brackets denote a variable requiring substitution for a real value.
Italic font	Used to denote the title of a book, article, or other publication.
Note	Additional information or caveats

Scoring Information

A scoring status indicates whether compliance with the given recommendation impacts the assessed target's benchmark score. The following scoring statuses are used in this benchmark:

Scored
Failure to comply with "Scored" recommendations will decrease the final benchmark score. Compliance with "Scored" recommendations will increase the final benchmark score.

Not Scored
Failure to comply with "Not Scored" recommendations will not decrease the final benchmark score. Compliance with "Not Scored" recommendations will not increase the final benchmark score.

Profile Definitions

The following configuration profiles are defined by this Benchmark:

- **Level 1 - IIS 8.0**

  Items in this profile apply to Microsoft IIS 8.0 and intend to:

  - be practical and prudent;
  - provide a clear security benefit; and
  - not inhibit the utility of the technology beyond acceptable means.

- **Level 2 - IIS 8.0**

  This profile extends the "Level 1 - IIS 8.0" profile. Items in this profile apply to Microsoft IIS 8.0 and exhibit one or more of the following characteristics:

  - are intended for environments or use cases where security is paramount
  - acts as defense in depth measure
  - may negatively inhibit the utility or performance of the technology.

- **Level 1 - IIS 8.5**

  Items in this profile apply to Microsoft IIS 8.5 and intend to:

  - be practical and prudent;
  - provide a clear security benefit; and
  - not inhibit the utility of the technology beyond acceptable means.

Acknowledgements

This benchmark exemplifies the great things a community of users, vendors, and subject matter experts can accomplish through consensus collaboration. The CIS community thanks the entire consensus team with special recognition to the following individuals who contributed greatly to the creation of this guide:

Editor
Terri Donahue
John Baker
Steve Schofield

Recommendations

1 Recommendations

1.1 Basic Configurations

This section contains basic Web server-level recommendations.

1.1.1 Ensure Web Content Is on Non-System Partition (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Web resources published through IIS are mapped, via Virtual Directories, to physical locations on disk. It is recommended to map all Virtual Directories to a non-system disk volume.

Rationale:

Isolating web content from system files may reduce the probability of:

- Web sites/applications exhausting system disk space
- File IO vulnerability in the web site/application from affecting the confidentiality and/or integrity of system files

Audit:

Execute the following command to ensure no virtual directories are mapped to the system drive:

```
%systemroot%\system32\inetsrv\appcmd list vdir
```

Remediation:

1. Browse to web content in `C:\inetpub\wwwroot\`
2. Copy or cut content onto a dedicated and restricted web folder on a non-system drive such as `D:\webroot\`
3. Change mappings for any applications or Virtual Directories to reflect the new location

To change the mapping for the application named app1 which resides under the Default Web Site, open IIS Manager:

1. Expand the server node
2. Expand Sites
3. Expand Default Web Site
4. Click on app1
5. In the Actions pane, select Basic Settings
6. In the Physical path text box, put the new location of the application, `D:\wwwroot\app1` in the example above

References:

1. http://technet.microsoft.com/en-us/library/cc179961.aspx
2. http://blogs.iis.net/thomad/archive/2008/02/10/moving-the-iis7-inetpub-directory-to-a-different-drive.aspx

1.1.2 Require Host Headers on all Sites (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

Host headers provide the ability to host multiple websites on the same IP address and port. It is recommended that host headers be configured for all sites.

Rationale:

Requiring a Host header for all sites may reduce the probability of:

- DNS rebinding attacks successfully compromising or abusing site data or functionality [2]
- IP-based scans successfully identifying or interacting with a target application hosted on IIS

Audit:

Execute the following command to identify sites that are not configured to require host headers:

```
%systemroot%\system32\inetsrv\appcmd list sites
```

All sites will be listed as such:

```
SITE "Default Web Site" (id:1,bindings:http/*:80:test.com,state:Started)
SITE "badsite" (id:3,bindings:http/*:80:,state:Started)
```

For all non-SSL sites, ensure that the *IP:port:host* binding triplet contains a host name. In the example above, the first site is configured as recommended given the `Default Web Site` has a host header of `test.com`. `badsite`, however, does not have a host header configured - it shows `*:80:` which means all IPs over port 80, with no host header.

Remediation:

Obtain a listing of all sites by using the following `appcmd.exe` command:

```
%systemroot%\system32\inetsrv\appcmd list sites
```

Perform the following in IIS Manager to configure host headers for the Default Web Site:

1. Open IIS Manager
2. In the Connections pane expand the Sites node and select Default Web Site
3. In the Actions pane click Bindings
4. In the Site Bindings dialog box, select the binding for which host headers are going to be configured, Port 80 in this example
5. Click Edit
6. Under host name, enter the sites FQDN, such as *<www.examplesite.com>*
7. Click OK, then Close

Note: Requiring a host header may impair site functionality for HTTP/1.0 clients.

References:

1. http://technet.microsoft.com/en-us/library/cc753195%28WS.10%29.aspx
2. http://crypto.stanford.edu/dns/dns-rebinding.pdf
3. http://www.sslshopper.com/article-ssl-host-headers-in-iis-7.html
4. http://blogs.iis.net/thomad/archive/2008/01/25/ssl-certificates-on-sites-with-host-headers.aspx

1.1.3 Disable Directory Browsing (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Directory browsing allows the contents of a directory to be displayed upon request from a web client. If directory browsing is enabled for a directory in Internet Information

Services, users receive a page that lists the contents of the directory when the following two conditions are met:

1. No specific file is requested in the URL
2. The Default Documents feature is disabled in IIS, or if it is enabled, IIS is unable to locate a file in the directory that matches a name specified in the IIS default document list

It is recommended that directory browsing be disabled.

Rationale:

Ensuring that directory browsing is disabled may reduce the probability of disclosing sensitive content that is inadvertently accessible via IIS.

Audit:

Perform the following to verify that Directory Browsing has been disabled at the server level:

```
%systemroot%\system32\inetsrv\appcmd list config /section:directoryBrowse
```

If the server is configured as recommended, the following will be displayed:

```
<system.webServer>
  <directoryBrowse enabled="false" />
<system.webServer>
```

Remediation:

Directory Browsing can be set by using the UI, running `appcmd.exe` commands, by editing configuration files directly, or by writing WMI scripts. To disable directory browsing at the server level using an `appcmd.exe` command:

```
%systemroot%\system32\inetsrv\appcmd set config /section:directoryBrowse
/enabled:false
```

References:

1. http://technet.microsoft.com/en-us/library/cc725840%28WS.10%29.aspx
2. http://technet.microsoft.com/en-us/library/cc731109%28WS.10%29.aspx

1.1.4 Set Default Application Pool Identity to Least Privilege Principal (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Application Pool Identities are the actual users/authorities that will run the worker process - `w3wp.exe`. Assigning the correct principal will help ensure that applications can function properly, while not giving overly permissive permissions on the system. These identities can further be used in ACLs to protect system content.

IIS 8.0 has additional built-in least privilege identities intended for use by Application Pools. It is recommended that the default Application Pool Identity be changed to a least privilege principle other than Network Service. Furthermore, it is recommended that all application pool identities be assigned a unique least privilege principal.

Rationale:

Setting Application Pools to use least privilege identities reduces the potential harm the identity could cause if the application becomes compromised.

Audit:

Execute the following command to determine if the `DefaultAppPool` identity has been changed to `ApplicationPoolIdentity`:

```
%systemroot%\system32\inetsrv\appcmd list config /section:applicationPools
```

Remediation:

The default Application Pool identity may be set for an application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, directly editing the configuration files, or by writing WMI scripts. Perform the following to change the default identity to the built-in `ApplicationPoolIdentity` in the IIS Manager GUI:

1. Open the IIS Manager GUI
2. In the connections pane, expand the server node and click Application Pools
3. On the Application Pools page, select the `DefaultAppPool`, and then click Advanced Settings in the Actions pane

4. For the Identity property, click the `'...'` button to open the Application Pool Identity dialog box
5. Select the Built-in account option choose `ApplicationPoolIdentity` from the list, or input a unique application user created for this purpose
6. Restart IIS

To change the `DefaultAppPool` identity to the built-in `ApplicationPoolIdentity` using `AppCmd.exe`, run the following from a command prompt:

```
%systemroot%\system32\inetsrv\appcmd set config /section:applicationPools
/[name='DefaultAppPool'].processModel.identityType:ApplicationPoolIdentity
```

If using a custom defined Windows user such as a dedicated service account, that user will need to be a member of the IIS_IUSRS group. The IIS_IUSRS group has access to all the necessary file and system resources so that an account, when added to this group, can seamlessly act as an application pool identity.

References:

1. http://technet.microsoft.com/en-us/library/cc771170%28WS.10%29.aspx
2. http://learn.iis.net/page.aspx/140/understanding-built-in-user-and-group-accounts-in-iis-7/

1.1.5 Ensure Application Pools Run Under Unique Identities (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Application Pool Identities are the actual users/authorities that will run the worker process - `w3wp.exe`. Assigning the correct user authority will help ensure that applications can function properly, while not giving overly permissive permissions on the system. These identities can further be used in ACLs to protect system content. It is recommended that each Application Pool run under a unique identity.

Rationale:

Setting Application Pools to use unique identities reduces the potential harm the identity could cause should the application become compromised.

Audit:

To verify the Application Pools have been set to run under the ApplicationPoolIdentity using IIS Manager:

1. Open IIS Manager
2. Open the Application Pools node underneath the machine node; select Application Pool to be verified
3. Right click the Application Pool and select Advanced Settings…
4. Under the Process Model section, locate the Identity option and ensure that `ApplicationPoolIdentity` is set

Remediation:

Setting the Application Pools to run under the `ApplicationPoolIdentity` will ensure that each pool runs under a unique authority. To configure the identity for the Application Pool, run the following `appcmd.exe` command from a command prompt:

```
%systemroot%\system32\inetsrv\appcmd set config /section:applicationPools
/[name='DefaultAppPool'].processModel.identityType:ApplicationPoolIdentity
```

The example code above will set just the `DefaultAppPool`. Run this command for each configured Application Pool. Additionally, `ApplicationPoolIdentity` can be made the default for all Application Pools by using the Set Application Pool Defaults action on the Application Pools node.

References:

1. http://technet.microsoft.com/en-us/library/cc753449%28WS.10%29.aspx
2. http://blogs.iis.net/tomwoolums/archive/2008/12/17/iis-7-0-application-pools.aspx
3. http://learn.iis.net/page.aspx/624/application-pool-identities/

1.1.6 Ensure Unique Application Pools for Sites (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

IIS 8.0 introduced a new security feature called Application Pool Identities that allows Application Pools to be run under unique accounts without the need to create and manage local or domain accounts. It is recommended that all Sites run under unique, dedicated Application Pools.

Rationale:

By setting sites to run under unique Application Pools, resource-intensive applications can be assigned to their own application pools which could improve server and application performance. In addition, it can help maintain application availability: if an application in one pool fails, applications in other pools are not affected. Last, isolating applications helps mitigate the potential risk of one application being allowed access to the resources of another application. It is also recommended to stop any application pool that is not in use or was created by an installation such as .Net 4.0.

Audit:

The following `appcmd.exe` command will give a listing of all applications configured, which site they are in, which application pool is serving them and which application pool identity they are running under:

```
%systemroot%\system32\inetsrv\appcmd list app
```

The output of this command will be similar to the following:

```
APP "Default Web Site/" (applicationPool:DefaultAppPool)
```

1. Run the above command and ensure a unique application pool is assigned for each site listed

Remediation:

1. Open IIS Manager
2. Open the Sites node underneath the machine node
3. Select the Site to be changed
4. In the Actions pane, select Basic Settings
5. Click the Select… box next to the Application Pool text box
6. Select the desired Application Pool
7. Once selected, click OK

References:

1. http://technet.microsoft.com/en-us/library/cc753449%28WS.10%29.aspx
2. http://blogs.iis.net/tomwoolums/archive/2008/12/17/iis-7-0-application-pools.aspx
3. http://learn.iis.net/page.aspx/624/application-pool-identities/

1.1.7 Configure Anonymous User Identity to Use Application Pool Identity (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

To achieve isolation in IIS 8, application pools can be run as separate identities. IIS can be configured to automatically use the application pool identity if no anonymous user account is configured for a Web site. This can greatly reduce the number of accounts needed for Web sites and make management of the accounts easier. It is recommended the Application Pool Identity be set as the Anonymous User Identity.

Rationale:

Configuring the anonymous user identity to use the application pool identity will help ensure site isolation - provided sites are set to use the application pool identity. Since a unique principal will run each application pool, it will ensure the identity is least privilege. Additionally, it will simplify Site management.

Audit:

Find and open the `applicationHost.config` file and verify that the `userName` attribute of the `anonymousAuthentication` tag is set to a blank string:

```
<system.webServer>
  <security>
    <authentication>
      <anonymousAuthentication userName="" />
    </authentication>
  </security>
</system.webServer>
```

This configuration is stored in the same `applicationHost.config` file for web sites and application/virtual directories, at the bottom of the file, surrounded by `<location path="path/to/resource">` tags.

Remediation:

The Anonymous User Identity can be set to Application Pool Identity by using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, directly editing the

configuration files, or by writing WMI scripts. Perform the following to set the username attribute of the `anonymousAuthentication` node in the IIS Manager GUI:

1. Open the IIS Manager GUI and navigate to the desired server, site, or application
2. In Features View, find and double-click the Authentication icon
3. Select the Anonymous Authentication option and in the Actions pane select Edit...
4. Choose Application pool identity in the modal window and then press the OK button

To use `AppCmd.exe` to configure `anonymousAuthentication` at the server level, the command would look like this:

```
%windir%\system32\inetsrv\appcmd set config -section:anonymousAuthentication
/username:"" --password
```

References:

1. http://learn.iis.net/page.aspx/202/application-pool-identity-as-anonymous-user/
2. http://learn.iis.net/page.aspx/624/application-pool-identities/

1.1.8 Configure Application Pools to Run As Application Pool Identity (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Application Pool Identities allow Application Pools to be run under a unique account without having to create and manage domain or local accounts. The name of the Application Pool account corresponds to the name of the Application Pool. Application Pool Identities were introduced in Windows Server 2008 SP2. It is recommended that Application Pools be set to run as `ApplicationPoolIdentity`. Note on applicable profiles: IIS 8.0 and IIS 8.5 only.

Rationale:

Setting Application Pools to use least privilege identities such as `ApplicationPoolIdentity` reduces the potential harm the identity could cause should the application become ever become compromised.

Audit:

Execute the following command to determine if the `DefaultAppPool` identity has been changed to `ApplicationPoolIdentity`:

```
%systemroot%\system32\inetsrv\appcmd list config /section:applicationPools
```

Remediation:

The default Application Pool identity may be set for an application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, directly editing the configuration files, or by writing WMI scripts. Perform the following to change the default identity to the built-in `ApplicationPoolIdentity` in the IIS Manager GUI:

1. Open the IIS Manager GUI
2. In the connections pane, expand the server node and click Application Pools
3. On the Application Pools page, select the `DefaultAppPool`, and then click Advanced Settings in the Actions pane
4. For the Identity property, click the `'. . .'` button to open the Application Pool Identity dialog box
5. Select the Built-in account option choose `ApplicationPoolIdentity` from the list
6. Restart IIS

To change the `DefaultAppPool` identity to the built-in `ApplicationPoolIdentity` using `AppCmd.exe`, run the following from a command prompt:

```
%systemroot%\system32\inetsrv\appcmd set config /section:applicationPools
/[name='<Your AppPool>'].processModel.identityType:ApplicationPoolIdentity
```

References:

1. http://learn.iis.net/page.aspx/624/application-pool-identities/

1.1.9 Use Only Strong Encryption Protocols (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

New and legacy web servers are often able and configured to handle weak cryptographic options due to historic export restriction of high grade cryptography. Even if high grade ciphers are normally used and installed, some server misconfiguration could be leveraged to force the use of a weaker protocol or cipher to gain access to the otherwise secure communication channel. It is recommended that the weak encryption protocols SSL 2.0 and PCT 1.0 be disabled, and that SSL 3.0 and TLS 1.x be enabled. It is recommended that you implement the **Diffie-Hellman Ephemeral**, or DHE for short, protocol for PFS.

Rationale:

SSL-based services should not offer the possibility to utilize weak encryption protocols or ciphers. By disabling these weaker protocols, data confidentiality and integrity is further assured.

Audit:

To verify the PCT 1.0 protocol is disabled on SP2, ensure the following key does not exist. If the key does exist, ensure it is set to `0`.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\PCT
1.0\Server\Enabled
```

To verify the PCT 1.0 protocol is disabled on R2, ensure the following key does not exist. If the key does exist, ensure it is set to `0`. PCT 1.0 is disabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\PCT
1.0\Server\DisabledByDefault
```

To verify the SSL 2.0 protocol is disabled on SP2 and R2, ensure the following key is set to `0`. SSL 2.0 is enabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL
2.0\Server\Enabled
```

To verify the SSL 3.0 protocol is enabled on R2 and SP2, ensure the following key does not exist. If the key does exist, ensure it is set to `ffffffff`. SSL 3.0 is enabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL
3.0\Server\Enabled
```

To verify the TLS 1.0 protocol is enabled on R2 and SP2, ensure the following key does not exist. If the key does exist, ensure it is set to `ffffffff`. TLS 1.0 is enabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS
1.0\Server\Enabled
```

To verify the TLS 1.1 protocol is enabled on R2, ensure the following key is set to `0`. At the time of this writing, TLS 1.1 is only supported on Windows Server 2008 R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS
1.1\Server\DisabledByDefault
```

To verify the TLS 1.2 protocol has been enabled on R2, ensure the following key is set to `0`. At the time of this writing, TLS 1.2 is only supported on Windows Server 2008 R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS
1.2\Server\DisabledByDefault
```

Remediation:

To disable the PCT 1.0 protocol on R2 and SP2, ensure the following key does not exist. If the key does exist, ensure it is set to `0`. PCT 1.0 is disabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\PCT
1.0\Server\Enabled
```

To disable the SSL 2.0 protocol on SP2 and R2, ensure the following key is set to `0`. SSL 2.0 is enabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL
2.0\Server\Enabled
```

To enable the SSL 3.0 protocol on R2 and SP2, ensure the following key does not exist. If the key does exist, ensure it is set to `ffffffff`. SSL 3.0 is enabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL
3.0\Server\Enabled
```

To enable the TLS 1.0 protocol on R2 and SP2, ensure the following key does not exist. If the key does exist, ensure it is set to `ffffffff`. TLS 1.0 is enabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS
1.0\Server\Enabled
```

To enable the TLS 1.1 protocol on R2, ensure the following key is set to `0`. At the time of this writing, TLS 1.1 is only supported on Windows Server 2008 R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS
1.1\Server\DisabledByDefault
```

To enable the TLS 1.2 protocol on R2, ensure the following key is set to `0`. At the time of this writing, TLS 1.2 is only supported on Windows Server 2008 R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS
1.2\Server\DisabledByDefault
```

References:

1. http://support.microsoft.com/kb/187498/en-us
2. http://support.microsoft.com/default.aspx?scid=kb;EN-US;245030
3. http://technet.microsoft.com/en-us/security/advisory/2588513
4. http://blogs.technet.com/b/erezs_iis_blog/archive/2013/08/22/perfect-secrecy-in-an-imperfect-world.aspx

1.1.10 Disable Weak Cipher Suites (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Both SSL 3.0 and TLS 1.0 provide options to use different cipher suites. Each cipher suite determines the key exchange, authentication, encryption, and MAC algorithms used within a SSL/TLS session. It is recommended that weak ciphers be disabled.

Rationale:

SSL-based services should not offer the possibility to utilize weak protocols or ciphers. By disabling these weak ciphers, there is a better chance of maintaining data confidentiality and integrity.

Audit:

To verify the `DES 56/56` cipher has been disabled, ensure the following key does not exist or is set to `0`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\DES 56/56
```

To verify the `NULL` cipher has been disabled, ensure the following key does not exist or is set to `0`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\NULL\Enabled
```

To verify the `RC2 40/128` cipher has been disabled, ensure the following key does not exist or is set to `0`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC2
40/128\Enabled
```

To verify the `RC2 56/128` cipher has been disabled, ensure the following key does not exist or is set to `0`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC2
56/128\Enabled
```

To verify the `RC4 40/128` cipher has been disabled, ensure the following key does not exist or is set to `0`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4
40/128\Enabled
```

To verify the `RC4 56/128` cipher has been disabled, ensure the following key does not exist or is set to `0`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4
56/128\Enabled
```

To verify the `RC4 64/128` cipher has been disabled, ensure the following key does not exist or is set to `0`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4
64/128\Enabled
```

To verify the `RC4 128/128` cipher has been disabled, ensure the following key either does not exist or is set to `0`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4
128/128\Enabled
```

To verify the `Triple DES 168/168` cipher has been enabled, ensure the following key either does not exist on R2 or is set to `ffffffff` on SP2 and R2:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\Triple DES
168/168\Enabled
```

To verify the `AES 256/256` cipher has been enabled on R2, ensure the following key is set to `ffffffff`:

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AES
256/256\Enabled
```

Remediation:

To disable weak ciphers perform all of the following and reboot the server when complete.

To disable `DES 56/56`, ensure the following key is absent. If the key is present, ensure it is set to `0`. The `DES 56/56` cipher is disabled by default on SP2 and R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\DES
56/56\Enabled
```

To disable the `NULL` cipher, ensure the following key is absent. If the key is present, ensure it is set to `0`. The `NULL` cipher is disabled by default on SP2 and R2.

```
HKLM System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\NULL\Enabled
```

To disable `RC2 40/128`, ensure the following key is absent. If the key is present, ensure it is set to `0`. The `RC2 40/128` cipher is disabled by default on SP2 and R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC2
40/128\Enabled
```

To disable `RC2 56/128`, ensure the following key is absent. If the key is present, ensure it is set to `0`. The `RC2 56/128` cipher is disabled by default on SP2 and R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC2
56/128\Enabled
```

To disable `RC4 40/128`, ensure the following key is absent. If the key is present, ensure it is set to `0`. The `RC4 40/128` cipher is disabled by default on R2 and SP2.

```
HKLM \System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4
40/128\Enabled
```

To disable `RC4 56/128`, ensure the following key is absent. If the key is present, ensure it is set to `0`. The `RC4 56/128` cipher is disabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4
56/128\Enabled
```

To disable `RC4 64/128`, ensure the following key is absent. If the key is present, ensure it is set to `0`. The `RC4 64/128` cipher is disabled by default on R2 and SP2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4
64/128\Enabled
```

To disable `RC4 128/128`, ensure the following key is absent. `If the key is present, ensure it is set to 0.`

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4
128/128\Enabled
```

To optionally enable strong ciphers, perform the following and reboot the server when complete:

To enable `Triple DES 168/168`, ensure the following key is set to `ffffffff`. The `Triple DES 168/168` cipher is not enabled by default on Server 2008 SP2 and is enabled by default on Server 2008 R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\Triple DES 168/168\Enabled
```

To enable `AES 256/256`, ensure the following key is set to `ffffffff`. The `AES 256/256` cipher is not supported on Server 2008 SP2 and is enabled by default on Server 2008 R2.

```
HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AES 256/256\Enabled
```

References:

1. https://www.owasp.org/index.php/Testing_for_SSL-TLS_%28OWASP-CM-001%29
2. http://support.microsoft.com/kb/245030/en-us
3. http://msdn.microsoft.com/en-us/library/aa374757%28v=vs.85%29.aspx
4. http://msdn.microsoft.com/en-us/library/aa380512%28v=vs.85%29.aspx

1.1.11 Enable FTP Logon Attempt Restrictions (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

IIS 8.0 introduced a built-in network security feature to automatically block brute force FTP attacks. This can be used to mitigate a malicious client from attempting a brute-force attack on a discovered account, such as the local administrator account.

Rationale:

Successful brute force FTP attacks can allow an otherwise unauthorized user to make changes to data that should not be made. This could allow the unauthorized user to modify website code by uploading malicious software or even changing functionality for items such as online payments.

Audit:

Access your FTP server using the administrator account and an invalid password. Verify after the maximum number of login attempts has been met that you receive a message 'Connection closed by remote host' when trying to access FTP.

Remediation:

1. Open IIS Manager

2. At the server level, open the FTP Logon Attempt Restrictions feature.

3. Check Enable FTP Logon Attempt Restrictions and enter the maximum number of failed attempts and the time period. Enable Deny IP addresses based on the number of failed login attempts.

4. Click Apply

Default Value:

By default, this feature is not enabled when FTP is installed.

References:

1. http://www.iis.net/learn/get-started/whats-new-in-iis-8/iis-80-ftp-logon-attempt-restrictions

1.1.12 Enable Dynamic IP Address Restrictions (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

IIS8 introduced the concept of Dynamic IP Address Restrictions which can be used to thwart DDos attacks. This is different than the IP Address Restrictions that can be manually maintained within IIS. The default action Deny action for restrictions is to return a Forbidden response to the client.

Rationale:

Dynamic IP address filtering allows administrators to configure the server to block access for IPs that exceed the specified number of requests. Ensure that you receive the Forbidden page once the block has been enforced.

Audit:

Access the web server enough times to trigger the IP restriction based on the settings entered.

Remediation:

1. Open IIS Manager.

2. Open the IP Address and Domain Restrictions feature.

3. Click Edit Dynamic Restrictions Settings..

4. Check the Deny IP Address based on the number of concurrent requests and the Deny IP Address based on the number of requests over a period of time boxes. The values can be tweaked as needed for your specific environment.

Default Value:

By default Dynamic IP Restrictions are not enabled.

References:

1. http://www.iis.net/learn/get-started/whats-new-in-iis-8/iis-80-dynamic-ip-address-restrictions

1.2 Configure Authentication

This section contains recommendations around the different layers of authentication in IIS.

1.2.1 Configure Global Authorization Rule to Restrict Access (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

IIS 7 introduced URL Authorization, which allows the addition of Authorization rules to the actual URL, instead of the underlying file system resource, as a way to protect it. Authorization rules can be configured at the server, web site, folder (including Virtual Directories), or file level. The native URL Authorization module applies to all requests, whether they are .NET managed or other types of files (e.g. static files or ASP files). It is recommended that URL Authorization be configured to only grant access to the necessary security principals.

Rationale:

Configuring a global Authorization rule that restricts access will ensure inheritance of the settings down through the hierarchy of web directories; if that content is copied elsewhere, the authorization rules flow with it. This will ensure access to current and future content is only granted to the appropriate principals, mitigating risk of accidental or unauthorized access.

Audit:

At the web site or application level, verify that the authorization rule configured has been applied:

1. Connect to Internet Information Services (IIS Manager)
2. Select the site or application where Authorization was configured
3. Select Authorization Rules and verify the configured rules were added

To verify an authorization rule specifying no access to all users except the `Administrators` group, browse to and open the `web.config` file for the configured site/application/content:

```
<configuration>
  <system.webServer>
    <security>
      <authorization>
        <remove users="*" roles="" verbs="" />
        <add accessType="Allow" roles="administrators" />
      </authorization>
    </security>
  </system.webServer>
</configuration>
```

Remediation:

To configure URL Authorization at the server level using IIS Manager:

1. Connect to Internet Information Services (IIS Manager)
2. Select the server
3. Select Authorization Rules
4. Remove the "Allow All Users" rule
5. Click Add Allow Rule…
6. Allow access to the user(s), user groups, or roles that are authorized across all of the web sites and applications (e.g. the Administratorsgroup)

References:

1. http://learn.iis.net/page.aspx/142/understanding-iis-70-url-authorization/
2. http://learn.iis.net/page.aspx/110/changes-between-iis6-and-iis7-security/

1.2.2 Ensure Access to Sensitive Site Features Is Restricted To Authenticated Principals Only (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

IIS 8 supports both challenge-based and login redirection-based authentication methods. Challenge-based authentication methods, such as Integrated Windows Authentication, require a client to respond correctly to a server-initiated challenge. A login redirection-based authentication method such as Forms Authentication relies on redirection to a login page to determine the identity of the principal. Challenge-based authentication and login redirection-based authentication methods cannot be used in conjunction with one another.

Public servers/sites are typically configured to use Anonymous Authentication. This method typically works, provided the content or services is intended for use by the public. When sites, applications, or specific content containers are not intended for anonymous public use, an appropriate authentication mechanism should be utilized. Authentication will help confirm the identity of clients who request access to sites, application, and content. IIS 7.0 provides the following authentication modules by default:

- Anonymous Authentication - allows anonymous users to access sites, applications, and/or content
- Integrated Windows Authentication - authenticates users using the NTLM or Kerberos protocols; Kerberos v5 requires a connection to Active Directory
- ASP.NET Impersonation - allows ASP.NET applications to run under a security context different from the default security context for an application
- Forms Authentication - enables a user to login to the configured space with a valid user name and password which is then validated against a database or other credentials store
- Basic authentication - requires a valid user name and password to access content
- Client Certificate Mapping Authentication - allows automatic authentication of users who log on with client certificates that have been configured; requires SSL
- Digest Authentication - uses Windows domain controller to authenticate users who request access

Note that none of the challenge-based authentication modules can be used at the same time Forms Authentication is enabled for certain applications/content. Forms Authentication does not rely on IIS authentication, so anonymous access for the ASP.NET application can be configured if Forms Authentication will be used.

It is recommended that sites containing sensitive information, confidential data, or non-public web services be configured with a credentials-based authentication mechanism.

Rationale:

Configuring authentication will help mitigate the risk of unauthorized users accessing data and/or services, and in some cases reduce the potential harm that can be done to a system.

Audit:

To verify that the authentication module is enabled for a specific site, application, or content, browse to and open the `web.config` file pertaining to the content. Verify the configuration file now has a mode defined within the `<authentication>`tags. The example below shows that Forms Authentication is configured, cookies will always be used, and SSL is required:

```
<system.web>
  <authentication>
    <forms cookieless="UseCookies" requireSSL="true" />
  </authentication>
</system.web>
```

Remediation:

Enabling authentication can be performed by using the user interface (UI), running `AppCmd.exe` commands in a command-line window, editing configuration files directly, or by writing WMI scripts. To verify an authentication mechanism is in place for sensitive content using the IIS Manager GUI:

1. Open IIS Manager and navigate to level with sensitive content
2. In Features View, double-click Authentication
3. On the Authentication page, make sure an authentication module is enabled, while anonymous authentication is enabled (Forms Authentication can have anonymous as well)
4. If necessary, select the desired authentication module, then in the Actions pane, click Enable

Note: When configuring an authentication module for the first time, each mechanism must be further configured before use.

References:

1. http://learn.iis.net/page.aspx/377/using-aspnet-forms-authentication/rev/1
2. http://learn.iis.net/page.aspx/244/how-to-take-advantage-of-the-iis7-integrated-pipeline/
3. http://technet.microsoft.com/en-us/library/cc733010%28WS.10%29.aspx
4. http://msdn.microsoft.com/en-us/library/aa480476.aspx

1.2.3 Require SSL in Forms Authentication (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Forms-based authentication can pass credentials across the network in clear text. It is therefore imperative that the traffic between client and server be encrypted using SSL, especially in cases where the site is publicly accessible. It is recommended that communications with any portion of a site using Forms Authentication be encrypted using SSL.

Rationale:

Requiring SSL for Forms Authentication will protect the confidentiality of credentials during the login process, helping mitigate the risk of stolen user information.

Audit:

To verify that SSL is required for forms authentication for a specific site, application, or content, browse to and open the `web.config` file for the level in which forms authentication was enabled. Verify the tag `<forms requireSSL="true" />`:

```
<system.web>
  <authentication>
    <forms requireSSL="true" />
  </authentication>
</system.web>
```

Remediation:

1. Open IIS Manager and navigate to the appropriate tier
2. In Features View, double-click Authentication
3. On the Authentication page, select Forms Authentication
4. In the Actions pane, click Edit
5. Check the Requires SSL checkbox in the cookie settings section, click OK

References:

1. http://technet.microsoft.com/en-us/library/cc771077(WS.10).aspx

1.2.4 Configure Forms Authentication to Use Cookies (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

Forms Authentication can be configured to maintain the site visitor's session identifier in either a URI or cookie. It is recommended that Forms Authentication be set to use cookies.

Rationale:

Using cookies to manage session state may help mitigate the risk of session hi-jacking attempts by preventing ASP.NET from having to move session information to the URL. Moving session information identifiers into the URL may cause session IDs to show up in proxy logs, browsing history, and be accessible to client scripting via `document.location`.

Audit:

Locate and open the `web.config` for the configured application. Verify the presence of `<forms cookieless="UseCookies" />`.

```
<system.web>
  <authentication>
    <forms cookieless="UseCookies" requireSSL="true" timeout="30" />
  </authentication>
</system.web>
```

Remediation:

1. Open IIS Manager and navigate to the level where Forms Authentication is enabled
2. In Features View, double-click Authentication
3. On the Authentication page, select Forms Authentication
4. In the Actions pane, click Edit
5. In the Cookie settings section, select Use cookies from the Mode dropdown

References:

1. http://technet.microsoft.com/en-us/library/cc732830%28WS.10%29.aspx

1.2.5 Configure Cookie Protection Mode for Forms Authentication (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The cookie protection mode defines the protection Forms Authentication cookies will be given within a configured application. The four cookie protection modes that can be defined are:

- Encryption and validation - Specifies that the application use both data validation and encryption to help protect the cookie; this option uses the configured data validation algorithm (based on the machine key) and triple-DES (3DES) for encryption, if available and if the key is long enough (48 bytes or more)
- None - Specifies that both encryption and validation are disabled for sites that are using cookies only for personalization and have weaker security requirements
- Encryption - Specifies that the cookie is encrypted by using Triple-DES or DES, but data validation is not performed on the cookie; cookies used in this manner might be subject to plain text attacks
- Validation - Specifies that a validation scheme verifies that the contents of an encrypted cookie have not been changed in transit

It is recommended that cookie protection mode always encrypt and validate Forms Authentication cookies.

Rationale:

By encrypting and validating the cookie, the confidentiality and integrity of data within the cookie is assured. This helps mitigate the risk of attacks such as session hijacking and impersonation.

Audit:

Locate and open the `web.config` for the configured application. Verify the presence of `<forms protection="All" />`.

```
<system.web>
  <authentication>
    <forms cookieless="UseCookies" protection="All" />
  </authentication>
</system.web>
```

Note: The `protection="All"` property will only show up if cookie protection mode was set to something different, and then changed to Encryption and validation. To truly verify the `protection="All"` property in the `web.config`, the protection mode can be changed, and then changed back. Conversely, the `protection="All"` line can be added to the `web.config` manually.

Remediation:

Cookie protection mode can be configured by using the user interface (UI), by running `Appcmd.exe` commands in a command-line window, by editing configuration files directly, or by writing WMI scripts. Using IIS Manager:

1. Open IIS Manager and navigate to the level where Forms Authentication is enabled
2. In Features View, double-click Authentication
3. On the Authentication page, select Forms Authentication
4. In the Actions pane, click Edit
5. In the Cookie settings section, verify the drop-down for Protection mode is set for Encryption and validation

References:

1. http://technet.microsoft.com/en-us/library/cc731804%28WS.10%29.aspx
2.

1.2.6 Ensure passwordFormat Credentials Element Not Set To Clear (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The `<credentials>` element of the `<authentication>` element allows optional definitions of name and password for IIS Manager User accounts within the configuration file. IIS Manager Users can use the administration interface to connect to sites and applications in which they've been granted authorization. Note that the `<credentials>` element only applies when the default provider, `ConfigurationAuthenticationProvider`, is configured as the authentication provider. It is recommended that `passwordFormat` be set to a value other than `Clear`, such as `SHA1` or `MD5`.

Rationale:

Authentication credentials should always be protected to reduce the risk of stolen authentication credentials.

Audit:

Locate and open the configuration file for the configured application. Verify the `passwordFormat` is not set to `Clear`:

```
<configuration>
  <system.web>
    <authentication mode="Forms">
      <forms name="SampleApp" loginUrl="/login.aspx">
        <credentials passwordFormat="SHA1">
          <user
            name="UserName1"
            password="SHA1EncryptedPassword1"/>
          <user
            name="UserName2"
            password="SHA1EncryptedPassword2"/>
        </credentials>
      </forms>
    </authentication>
  </system.web>
</configuration>
```

Remediation:

Authentication mode is configurable at the `machine.config`, root-level `web.config`, or application-level `web.config`:

1. Locate and open the configuration file where the credentials are stored
2. Find the `<credentials>` element
3. If present, ensure `passwordFormat` is not set to `Clear`
4. Change `passwordFormat` to `SHA1` or `MD5`

The clear text passwords will need to be replaced with the appropriate hashed version.

References:

1. http://msdn.microsoft.com/en-us/library/e01fc50a.aspx
2. http://www.iis.net/ConfigReference/system.webServer/management/authentication/credentials
3. http://msdn.microsoft.com/en-us/library/bb422401%28VS.90%29.aspx

1.2.7 Lock down Encryption Providers (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

By default, whenever a property is encrypted, IIS 8.0 uses the `defaultProvider` for encryption defined in `machine.config`. The IIS 8.0 local system process (WAS) runs under the context of `LOCALSYSTEM` and needs access to the application pool passwords. However, by default the `IIS_IUSRS` security group is granted read access. It is recommended that the `IIS_IUSRS` group have access to the `iisWasKey` revoked.

Rationale:

The `iisWasKey` is intended for access only by Administrators and `SYSTEM`. Since the `IIS_IUSRS` group is granted read access, an attacker compromising an application set to use a principal in the `IIS_IUSRS` group could potentially gain access to the encryption key(s). Revoking this unnecessary privilege will reduce attack surface and help maintain confidentiality and system/application integrity.

Audit:

To verify the permissions have been removed:

1. Obtain the machine GUID at the Registry Value "MachineGuid" in the Registry Key: `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography`
2. Next, open a command prompt and run the following `icacls` command, ensuring that `BUILTIN\IIS_IUSRS(R)` has been removed:

```
icacls
%ALLUSERSPROFILE%\Microsoft\Crypto\RSA\MachineKeys\76944fb33636aeddb9590521c2e8815a_<M
achineGUID>
```

Remediation:

Removing access to the `iisWasKey` can be done by using an `aspnet_regiis.exe` command. The syntax is as follows, and is dependent on the version of .NET being used:

```
%systemroot%\Microsoft.NET\Framework\\aspnet_regiis.exe -pr iisWasKey IIS_IUSRS
```

To remove read access to the `IIS_IUSRS` security group on a system using .NET Framework v2.0:

1. Open an elevated command prompt
2. Run the following `aspnet_regiis.exe` command:

```
%systemroot%\Microsoft.NET\Framework\v2.0.50727\aspnet_regiis.exe -pr iisWasKey
IIS_IUSRS
```

If running a 64-bit system, also run the following:

```
%systemroot%\Microsoft.NET\Framework64\v2.0.50727\aspnet_regiis.exe -pr iisWasKey
IIS_IUSRS
```

Note: A unique version of `aspnet_regiis.exe` is included with each version of the .NET Framework. Since each version of the tool applies only to its associated version of the .NET Framework, be sure to use the appropriate version of the tool.

References:

1. http://learn.iis.net/page.aspx/141/using-encryption-to-protect-passwords/
2. http://support.microsoft.com/kb/977754

1.2.8 Configure SSL for Basic Authentication (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Basic Authentication can pass credentials across the network in clear text. It is therefore imperative that the traffic between client and server be encrypted using SSL, especially in cases where the site is publicly accessible and is recommended that SSL be configured and required for any Site or Application using Basic Authentication.

Rationale:

Credentials sent in clear text can be easily intercepted by malicious code or persons. Enforcing the use of Secure Sockets Layer will help mitigate the chances of hijacked credentials.

Audit:

Once SSL has been configured and required for a Site or application, only the https:// address will be available. Attempt loading the Site or application for which Basic Authentication is configured using http://, the requests will fail and IIS will throw a 403.4 - Forbidden error.

Remediation:

To Use Basic Authentication with SSL:

1. Open IIS Manager
2. In the Connections pane on the left, select the server to be configured
3. In the Connections pane, expand the server, then expand Sites and select the site to be configured

4. In the Actions pane, click Bindings; the Site Bindings dialog appears
5. If an HTTPS binding is available, click Close and see below "To require SSL"
6. If no HTTPS binding is visible, perform the following steps

To add an HTTPS binding:

1. In the Site Bindings dialog, click Add; the Add Site Binding dialog appears
2. Under Type, select https
3. Under SSL certificate, select an SSL certificate
4. Click OK, then close

To require SSL:

1. In Features View, double-click SSL Settings
2. On the SSL Settings page, select Require SSL, and Require 128-bit SSL
3. In the Actions pane, click Apply

References:

1. http://technet.microsoft.com/en-us/library/dd378853%28WS.10%29.aspx

1.3 ASP.NET Configuration Recommendations

This section contains recommendations specific to ASP.NET.

1.3.1 Set Deployment Method to Retail (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The `<deployment retail>` switch is intended for use by production IIS servers. This switch is used to help applications run with the best possible performance and least possible security information leakages by disabling the application's ability to generate trace output on a page, disabling the ability to display detailed error messages to end users, and disabling the debug switch. Often times, switches and options that are developer-focused, such as failed request tracing and debugging, are enabled during active development. It is recommended that the deployment method on any production server be set to `retail`.

Rationale:

Utilizing the switch specifically intended for production IIS servers will eliminate the risk of vital application and system information leakages that would otherwise occur if tracing or debug were to be left enabled, or `customErrors` were to be left off.

Audit:

After the next time IIS is restarted, open the `machine.config` file and verify that `<deployment retail="true" />` remains set to `true`.

```
<system.web>
  <deployment retail="true" />
</system.web>
```

Remediation:

1. Open the `machine.config` file located in: `%windir%\Microsoft.NET\Framework\<framework_version>\CONFIG`
2. Add the line `<deployment retail="true" />` within the `<system.web>` section
3. If systems are 64-bit, do the same for the `machine.config` located in: `%windir%\Microsoft.NET\Framework64\<framework_version>\CONFIG`

References:

1. http://msdn.microsoft.com/en-US/library/ms228298%28VS.80%29.aspx

1.3.2 Turn Debug Off (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

Developers often enable the debug mode during active ASP.NET development so that they do not have to continually clear their browsers cache every time they make a change to a resource handler. The problem would arise from this being left "on" or set to "true". Compilation debug output is displayed to the end user, allowing malicious persons to obtain detailed information about applications.

This is a defense in depth recommendation due to the `<deployment retail="true" />` in the `machine.config` configuration file overriding any debug settings. It is recommended that debugging still be turned off.

Rationale:

Setting `<compilation debug>` to `false` ensures that detailed error information does not inadvertently display during live application usage, mitigating the risk of application information leakage falling into unscrupulous hands.

Audit:

Browse to and open the `web.config` file pertaining to the server or specific application that has been configured. Locate the `<compilation debug>`switch and verify it is set to false.

```
<configuration>
  <system.web>
    <compilation debug="false" />
  </system.web>
</configuration>
```

Remediation:

To use the UI to make this change:

1. Open IIS Manager and navigate desired server, site, or application
2. In Features View, double-click .NET Compilation
3. On the .NET Compilation page, in the Behavior section, ensure the Debug field is set to False
4. When finished, click Apply in the Actions pane

Note: The `<compilation debug>` switch will not be present in the `web.config` file unless it has been added manually, or has previously been configured using the IIS Manager GUI.

References:

1. http://technet.microsoft.com/en-us/library/cc725812%28WS.10%29.aspx

1.3.3 Ensure Custom Error Messages are not Off (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

When an ASP.NET application fails and causes an HTTP/1.x 500 Internal Server Error, or a feature configuration (such as Request Filtering) prevents a page from being displayed, an error message will be generated. Administrators can choose whether or not the application should display a friendly message to the client, detailed error message to the

client, or detailed error message to localhost only. The `<customErrors>` tag in the `web.config` has three modes:

- On: Specifies that custom errors are enabled. If no `defaultRedirect` attribute is specified, users see a generic error. The custom errors are shown to the remote clients and to the local host
- Off: Specifies that custom errors are disabled. The detailed ASP.NET errors are shown to the remote clients and to the local host
- RemoteOnly: Specifies that custom errors are shown only to the remote clients, and that ASP.NET errors are shown to the local host. This is the default value

This is a defense in depth recommendation due to the `<deployment retail="true" />` in the `machine.config` file overriding any settings for `customErrors` to be turned `Off`. It is recommended that `customErrors` still be turned to `On` or `RemoteOnly`.

Rationale:

`customErrors` can be set to `On` or `RemoteOnly` without leaking detailed application information to the client. Ensuring that `customErrors` is not set to `Off` will help mitigate the risk of malicious persons learning detailed application error and server configuration information.

Audit:

Find and open the `web.config` file for the application/site and verify that the tag has either `<customErrors mode="RemoteOnly" />` or `<customErrors mode="On" />` defined.

Remediation:

`customErrors` may be set for a server, site, or application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, directly editing the configuration files, or by writing WMI scripts. Perform the following to set the `customErrors` mode to `RemoteOnly` or `On` for a Web Site in the IIS Manager GUI:

1. Open the IIS Manager GUI and navigate to the site to be configured
2. In Features View, find and double-click .NET Error Pages icon
3. In the Actions Pane, click Edit Feature Settings
4. In modal dialog, choose On or Remote Only for Mode settings
5. Click OK

References:

1. http://msdn.microsoft.com/en-us/library/h0hfz6fc.aspx
2. http://technet.microsoft.com/en-us/library/dd569096%28WS.10%29.aspx

1.3.4 ASP.NET stack tracing is Not Enabled (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

The `trace` element configures the ASP.NET code tracing service that controls how trace results are gathered, stored, and displayed. When tracing is enabled, each page request generates trace messages that can be appended to the page output or stored in an application trace log.

This is a defense in depth recommendation due to the `<deployment retail="true" />` in the `machine.config` file overriding any settings for ASP.NET stack tracing that are left on. It is recommended that ASP.NET stack tracing still be turned off.

Rationale:

In an active Web Site, tracing should not be enabled because it can display sensitive configuration and detailed stack trace information to anyone who views the pages in the site. If necessary, the `localOnly` attribute can be set to true to have trace information displayed only for localhost requests. Ensuring that ASP.NET stack tracing is not on will help mitigate the risk of malicious persons learning detailed stack trace information.

Audit:

Verify asp.net tracing is not turned on, via a per-page basis in the application.

Ensure the trace attribute is not enabled like:

```
Trace="true"
```

On an application basis like in the web.config ensure that tracing is not enabled like:

```
<configuration>
  <system.web>
  ...
    <trace enabled="true">
  ...
  </system.web>
</configuration
```

Note that tracing is configurable at numerous levels:
1. Machine.config
2. Root-level web.config

3. Application-level web.config
4. Virtual or physical directory-level web.config

Remediation:

1) ensure `<deployment retail="true" /> is` enabled in the machine.config.

2) Remove all attribute references to ASP.NET tracing by deleting the trace and trace enable attributes.

Per Page:

Remove any references to

```
Trace="true"
```

Per Application:

Remove any references to:

```
<configuration>
  <system.web>
  ...
    <trace enabled="true">
  ...
  </system.web>
</configuration>
```

Default Value:

The default value for ASP.NET tracing is off.

References:

1. http://msdn.microsoft.com/en-us/library/94c55d08%28v=vs.100%29.aspx
2. http://msdn.microsoft.com/en-us/library/0x5wc973%28v=vs.100%29.aspx

1.3.5 Configure Use Cookies Mode for Session State (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

A session cookie associates session information with client information for that session, which can be the duration of a user's connection to a site. The cookie is passed in a HTTP header together with all requests between the client and server.

Session information can also be stored in the URL. However, storing session information in this manner has security implications that can open attack vectors such as session hijacking. An effective method used to prevent session hijacking attacks is to force web applications to use cookies to store the session token. This is accomplished by setting the `cookieless` attribute of the `sessionState` node to `UseCookies` or `False` which will in turn keep session state data out of URI. It is recommended that session state be configured to `UseCookies`.

Rationale:

Cookies that have been properly configured help mitigate the risk of attacks such as session hi-jacking attempts by preventing ASP.NET from having to move session information to the URL; moving session information in URI causes session IDs to show up in proxy logs, and is accessible to client scripting via `document.location`.

Audit:

Find and open the `web.config` file for the application/site and verify that the `sessionState` tag is set to use cookies:

```
<system.web>
  <sessionState cookieless="UseCookies" />
</system.web>
```

Remediation:

`SessionState` can be set to `UseCookies` by using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, directly editing the configuration files, or by writing WMI scripts. Perform the following to set the `cookieless` attribute of the `sessionState` node to `UseCookies` in the IIS Manager GUI:

1. Open the IIS Manager GUI and navigate desired server, site, or application
2. In Features View, find and double-click the Session State icon
3. In the Cookie Settings section, choose Use Cookies from the Mode dropdown
4. In the Actions Pane, click Apply

To use `AppCmd.exe` to configure `sessionState` at the server level, the command would look like this:

```
%windir%\system32\inetsrv\appcmd set config /commit:WEBROOT /section:sessionState
/cookieless:UseCookies /cookieName:ASP.NET_SessionID /timeout:20
```

Note: When `Appcmd.exe` is used to configure the `<sessionstate>` element at the global level in IIS 8.0, the `/commit:WEBROOT` switch must be included so that configuration changes are made to the root `web.config` file instead of `ApplicationHost.config`.

References:

1. http://technet.microsoft.com/en-us/library/cc770672%28WS.10%29.aspx
2. http://msdn.microsoft.com/en-us/library/h6bb9cz9%28VS.71%29.aspx

1.3.6 Ensure Cookies Are Set With HttpOnly Attribute (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

The `httpOnlyCookies` attribute of the `httpCookies` node determines if IIS will set the `HttpOnly` flag on HTTP cookies it sets. The `HttpOnly` flag indicates to the user agent that the cookie must not be accessible by client-side script (i.e document.cookie). It is recommended that the `httpOnlyCookies` attribute be set to `true`.

Rationale:

When cookies are set with the `HttpOnly` flag, they cannot be accessed by client side scripting running in the user's browser. Preventing client-side scripting from accessing cookie content may reduce the probability of a cross site scripting attack materializing into a successful session hijack.

Audit:

After the next time IIS is restarted, browse to and open the `web.config` for the application in which `httpOnly` cookies have been turned on. Confirm the `httpOnlyCookies` attribute is set to true: `<httpCookies httpOnlyCookies="true" />`.

Remediation:

1. Locate and open the application's `web.config` file
2. Add the `<httpCookies httpOnlyCookies="true" />` tag within `<system.web>`:

```
<configuration>
  <system.web>
    <httpCookies httpOnlyCookies="true" />
  </system.web>
</configuration>
```

Note: Setting the value of the `httpOnlyCookies` attribute of the `httpCookies` element to `true` will add the `HttpOnly` flag to all the cookies set by the application. This attribute, `HttpOnly`, is currently only recognized by modern versions of Internet Explorer; older versions will either treat them as normal cookies or simply ignore them altogether.

References:

1. http://authors.aspalliance.com/aspxtreme/aspnet/syntax/httpCookies.aspx
2. http://www.asp101.com/tips/index.asp?id=160
3. https://tools.ietf.org/wg/httpstate/charters

1.3.7 Configure MachineKey Validation Method (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The `machineKey` element of the ASP.NET `web.config` specifies the algorithm and keys that ASP.NET will use for encryption. The Machine Key feature can be managed to specify hashing and encryption settings for application services such as view state, Forms authentication, membership and roles, and anonymous identification.

The following encryption methods are available:

- Advanced Encryption Standard (AES) is relatively easy to implement and requires little memory. AES has a key size of 128, 192, or 256 bits. This method uses the same private key to encrypt and decrypt data, whereas a public-key method must use a pair of keys
- Message Digest 5 (MD5) is used for digital signing of applications. This method produces a 128-bit message digest, which is a compressed form of the original data
- Secure Hash Algorithm (SHA1) is considered more secure than MD5 because it produces a 160-bit message digest
- Triple Data Encryption Standard (TripleDES) is a minor variation of Data Encryption Standard (DES). It is three times slower than regular DES but can be more secure because it has a key size of 192 bits. If performance is not a primary consideration, consider using TripleDES

It is recommended that AES or SHA1 methods be configured for use at the global level.

Rationale:

Setting the validation property to AES will provide confidentiality and integrity protection to the viewstate. AES is the strongest encryption algorithm supported by the validation property. Setting the validation property to SHA1 will provide integrity protection to the viewstate. SHA1 is the strongest hashing algorithm supported by the validation property.

Audit:

To verify the Machine Key encryption method using IIS Manager:

1. Open IIS Manager and navigate to the level that was configured, the WEBROOT, or server in this case
2. In the features view, double click Machine Key
3. On the Machine Key page, verify that SHA1 is selected in the Encryption method dropdown

Remediation:

Machine key encryption can be set by using the UI, running `appcmd.exe` commands, by editing configuration files directly, or by writing WMI scripts. To set the Machine Key encryption at the global level using an `appcmd.exe` command:

```
%systemroot%\system32\inetsrv\appcmd set config /commit:WEBROOT /section:machineKey
/validation:SHA1
```

Note: When `Appcmd.exe` is used to configure the `<machineKey>` element at the global level in IIS 7.0, the `/commit:WEBROOT` switch must be included so that configuration changes are made to the root `web.config` file instead of `ApplicationHost.config`.

References:

1. http://technet.microsoft.com/en-us/library/cc772271%28WS.10%29.aspx
2. http://technet.microsoft.com/en-us/library/cc772287%28WS.10%29.aspx

1.3.8 Configure Global .NET Trust Level (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

This only applies to .Net 2.0. Future versions have stopped supporting this feature.

An application's trust level determines the permissions that are granted by the ASP.NET code access security (CAS) policy. CAS defines two trust categories: full trust and partial trust. An application that has full trust permissions may access all resource types on a server and perform privileged operations, while applications that run with partial trust have varying levels of operating permissions and access to resources.

The possible values for the Level property of the TrustSection class are:

- Full: Specifies unrestricted permissions and grants the ASP.NET application permissions to access any resource that is subject to operating system security; all privileged operations are supported

- High: specifies a high level of code access security which limits the application from doing the following:
    - Call unmanaged code
    - Call serviced components
    - Write to the event log
    - Access Microsoft Windows Message Queuing queues
    - Access ODBC, OLD DB, or Oracle data sources
- Medium: specifies a medium level of code access security, which means that in addition to the restrictions for High, the ASP.NET application cannot do any of the following things:
    - Access files outside the application directory
    - Access the registry
- Low: specifies a low level of code access security, which means that in addition to the restrictions for Medium, the application is prevented from performing any of the following actions:
    - Write to the file system
    - Call the `System.Security.CodeAccessPermission.Assert` method to expand permissions to resources
    - Minimal: specifies a minimal level of code access security, which means that the application has only execute permission

It is recommended that the global .NET Trust Level be set to Medium or lower.

Rationale:

The CAS determines the permissions that are granted to the application on the server. Setting a minimal level of trust that is compatible with the applications will limit the potential harm that a compromised application could cause to a system.

Audit:

To verify the global .NET Trust Level using IIS Manager:

1. Open IIS Manager and navigate to the level that was configured, the server in this example
2. In the features view, double click .NET Trust Levels
3. On the .NET Trust Levels page, verify that `Medium (web_mediumtrust.config)` is selected in the Trust Level dropdown

Remediation:

Trust level can be set by using the UI, running `appcmd.exe` commands, by editing configuration files directly, or by writing WMI scripts. To set the .Net Trust Level to Medium at the server level using an `appcmd.exe` command:

```
%systemroot%\system32\inetsrv\appcmd set config /commit:WEBROOT /section:trust
/level:Medium
```

Note: When `Appcmd.exe` is used to configure the element at the global level in IIS 7.0, the `/commit:WEBROOT` switch must be included so that configuration changes are made to the root `web.config` file instead of `ApplicationHost.config`.

References:

1. http://technet.microsoft.com/en-us/library/cc772237(WS.10).aspx
2. http://msdn.microsoft.com/en-us/library/ms691448%28VS.90%29.aspx
3. Professional IIS 7 by Ken Schaefer, Jeff Cochran, Scott Forsyth, Rob Baugh, Mike Everest, Dennis Glendenning
4. http://support.microsoft.com/kb/2698981

1.3.9 Hide IIS HTTP Detailed Errors from Displaying Remotely (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

A Web site's error pages are often set to show detailed error information for troubleshooting purposes during testing or initial deployment. To prevent unauthorized users from viewing this privileged information, detailed error pages must not be seen by remote users. This setting can be modified in the `errorMode` attribute setting for a Web site's error pages. By default, the `errorMode` attribute is set in the `Web.config` file for the Web site or application and is located in the `<httpErrors>` element of the `<system.webServer>` section. It is recommended that custom errors be prevented from displaying remotely.

Rationale:

The information contained in custom error messages can provide clues as to how applications function, opening up unnecessary attack vectors. Ensuring custom errors are never displayed remotely can help mitigate the risk of malicious persons obtaining information as to how the application works.

Audit:

The `errorMode` attribute is set in the `Web.config` file for the Web site or application in the `<httpErrors>` element of the `<system.webServer>` section. Browse to the `web.config` and verify the `errorMode` is set to `DetailedLocalOnly` or `Custom`:

```
<system.web>
  <system.webServer>
    <httpErrors errorMode="DetailedLocalOnly">
    </httpErrors>
  </system.webServer>
</system.web>
```

Remediation:

The following describes how to change the `errorMode` attribute to `DetailedLocalOnly` or `Custom` for a Web site by using IIS Manager:

1. Open IIS Manager with Administrative privileges
2. In the Connections pane on the left, expand the server, then expand the Sites folder
3. Select the Web site or application to be configured
4. In Features View, select Error Pages, in the Actions pane, select Open Feature
5. In the Actions pane, select Edit Feature Settings
6. In the Edit Error Pages Settings dialog, under Error Responses, select either Custom error pages or Detailed errors for local requests and custom error pages for remote requests
7. Click OK and exit the Edit Error Pages Settings dialog

References:

1. http://technet.microsoft.com/en-us/library/dd391900%28WS.10%29.aspx
2. http://www.iis.net/configreference/system.webserver/httperrors

1.4 Request Filtering and Restrictions

Request Filtering is a powerful new module implemented in IIS 7.0 which provides a configurable set of rules that enables administrators to allow or reject the types of requests that they determine should be allowed or rejected at the server, web site, or web application levels.

Earlier versions of Internet Information Services provided the tool UrlScan, which was provided as an add-on to enable system administrators to enforce tighter security policies on their web servers. For IIS 7.0, all of the core features of URLScan have been incorporated into the Request Filtering module. Due to the close nature of functionality in these two tools, reference to legacy URLScan settings will be made where applicable.

Note: Request Filtering must be installed as a role service under IIS in order to configure any of its features. Additionally, to edit Request Filtering settings with the IIS Management GUI requires the IIS 7.0 Administration Pack, which is provided as an IIS Extension and available on the Web.

1.4.1 Configure maxAllowedContentLength Request Filter (Not Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

The `maxAllowedContentLength` Request Filter is the maximum size of the http request, measured in bytes, which can be sent from a client to the server. Configuring this value enables the total request size to be restricted to a configured value. It is recommended that the overall size of requests be restricted to a maximum value appropriate for the server, site, or application.

Rationale:

Setting an appropriate value that has been tested for the `maxAllowedContentLength` filter will lower the impact an abnormally large request would otherwise have on IIS and/or web applications. This helps to ensure availability of web content and services, and may also help mitigate the risk of buffer overflow type attacks in unmanaged components.

Audit:

Upon exceeding the configured value set for the Request Filter, IIS will throw a Status Code 404.13.

To manually verify the change, locate and open the `web.config` for the web site or application in which the request filter was set. Ensure the value defined for `maxAllowedContentLength` is what was set. The 28.6MB max example would show:

```
<configuration>
  <system.webServer>
    <security>
      <requestFiltering>
        <requestLimits
           maxAllowedContentLength="30000000" />
      </requestFiltering>
    </security>
  </system.webServer>
</configuration>
```

Remediation:

The `MaxAllowedContentLength` Request Filter may be set for a server, website, or application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, and/or directly editing the configuration files. To configure using the IIS Manager GUI:

1. Open Internet Information Services (IIS) Manager
2. In the Connections pane, click on the server, site, application, or directory to be configured
3. In the Home pane, double-click Request Filtering
4. Click Edit Feature Settings... in the Actions pane
5. Under the Request Limits section, key the maximum content length in bytes that will allow applications to retain their intended functionality, such as 30000000 (approx. 28.6 MB)

References:

1. http://www.iis.net/ConfigReference/system.webServer/security/requestFiltering/requestLimits
2. http://learn.iis.net/page.aspx/143/use-request-filtering/

1.4.2 Configure maxURL Request Filter (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

The `maxURL` attribute of the `<requestLimits>` property is the maximum length (in Bytes) in which a requested URL can be (excluding query string) in order for IIS to accept. Configuring this Request Filter enables administrators to restrict the length of the requests that the server will accept. It is recommended that a limit be put on the length of URI.

Rationale:

With a properly configured Request Filter limiting the amount of data accepted in the URL, chances of undesired application behaviors affecting the availability of content and services are reduced.

Audit:

IIS will log a 404.14 HTTP status if the requested URL was rejected because it exceeded the length defined in the filter.

To manually verify the change, locate and open the `web.config` for the web site or application in which the request filter was set. Verify the value defined for maxURL.

```
<configuration>
  <system.webServer>
    <security>
      <requestFiltering>
        <requestLimits
           maxURL="4096" />
      </requestFiltering>
    </security>
  </system.webServer>
</configuration>
```

Remediation:

The `MaxURL` Request Filter may be set for a server, website, or application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, and/or directly editing the configuration files. To configure using the IIS Manager GUI:

1. Open Internet Information Services (IIS) Manager
2. In the Connections pane, click on the connection, site, application, or directory to be configured
3. In the Home pane, double-click Request Filtering
4. Click Edit Feature Settings... in the Actions pane
5. Under the Request Limits section, key the maximum URL length in bytes that has been tested with web applications

References:

1. http://www.iis.net/ConfigReference/system.webServer/security/requestFiltering/requestLimits
2. http://learn.iis.net/page.aspx/143/use-request-filtering/

1.4.3 Configure MaxQueryString Request Filter (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

The `MaxQueryString` Request Filter describes the upper limit on the length of the query string that the configured IIS server will allow for websites or applications. It is recommended that values always be established to limit the amount of data will can be accepted in the query string.

Rationale:

With a properly configured Request Filter limiting the amount of data accepted in the query string, chances of undesired application behaviors such as app pool failures are reduced.

Audit:

If a request is rejected because it exceeds the value set in the `maxQueryString` Request Filter, a 404.15 HTTP status is logged to the IIS log file.

To manually verify the change, locate and open `the web.config` for the web site or application in which the filter was set. Ensure the value defined for `maxQueryString` is what was configured.

```
<configuration>
  <system.webServer>
    <security>
      <requestFiltering>
        <requestLimits
           maxQueryString="2048" />
      </requestFiltering>
    </security>
  </system.webServer>
</configuration>
```

Remediation:

The `MaxQueryString` Request Filter may be set for a server, website, or application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, and/or directly editing the configuration files. To configure using the IIS Manager GUI:

1. Open Internet Information Services (IIS) Manager
2. In the Connections pane, go to the connection, site, application, or directory to be configured
3. In the Home pane, double-click Request Filtering
4. Click Edit Feature Settings... in the Actions pane
5. Under the Request Limits section, key in a safe upper bound in the Maximum query string (Bytes) textbox

References:

1. http://www.iis.net/ConfigReference/system.webServer/security/requestFiltering/requestLimits
2. http://learn.iis.net/page.aspx/143/use-request-filtering/

1.4.4 Disallow non-ASCII Characters in URLs (Scored)

Profile Applicability:

- Level 2 - IIS 8.0

Description:

This feature is used to allow or reject all requests to IIS 7 that contain non-ASCII characters. When using this feature, Request Filtering will deny the request if high-bit characters are present in the URL. The UrlScan equivalent is `AllowHighBitCharacters`. It is recommended that requests containing non-ASCII characters be rejected, where possible.

Rationale:

This feature can help defend against canonicalization attacks, reducing the potential attack surface of servers, sites, and/or applications.

Audit:

If a request is rejected because it contains a high-bit character, a 404.12 HTTP status is logged to the IIS log file.

To manually verify the change, locate and open the `web.config` for the web site or application in which the request filter was set. Ensure the value defined for the filter is false, as such:

```
<configuration>
  <system.webServer>
    <security>
      <requestFiltering
         allowHighBitCharacters="false">
      </requestFiltering>
    </security>
  </system.webServer>
</configuration>
```

Remediation:

The `AllowHighBitCharacters` Request Filter may be set for a server, website, or application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, and/or directly editing the configuration files. To configure using the IIS Manager GUI:

1. Open Internet Information Services (IIS) Manager
2. In the Connections pane, go to the connection, site, application, or directory to be configured
3. In the Home pane, double-click Request Filtering

4. Click Edit Feature Settings... in the Actions pane
5. Under the General section, uncheck Allow high-bit characters

Note: Disallowing high-bit ASCII characters in the URL may negatively impact the functionality of sites requiring international language support.

References:

1. http://learn.iis.net/page.aspx/143/use-request-filtering/
2. http://learn.iis.net/page.aspx/936/urlscan-1-reference/
3. Professional IIS 7 by Ken Schaefer, Jeff Cochran, Scott Forsyth, Rob Baugh, Mike Everest, Dennis Glendenning

1.4.5 Ensure Double-Encoded Requests will be Rejected (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

This Request Filter feature prevents attacks that rely on double-encoded requests and applies if an attacker submits a double-encoded request to IIS. When the double-encoded requests filter is enabled, IIS 7 will go through a two iteration process of normalizing the request. If the first normalization differs from the second, the request is rejected and the error code is logged as a 404.11. The double-encoded requests filter was the `VerifyNormalization` option in UrlScan. It is recommended that double-encoded requests be rejected.

Rationale:

This feature will help prevent attacks that rely on URLs that have been crafted to contain double-encoded request(s).

Audit:

If a request is rejected because it contains a double-encoded request, a 404.11 HTTP status is logged to the IIS log file.

To manually verify the change, locate and open the `web.config` for the web site or application in which the request filter was set. Ensure the value defined for `allowDoubleEscaping` is `false`:

```
<configuration>
  <system.webServer>
    <security>
      <requestFiltering
         allowDoubleEscaping="false">
      </requestFiltering>
    </security>
  </system.webServer>
</configuration>
```

Remediation:

The `allowDoubleEscaping` Request Filter may be set for a server, website, or application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, and/or directly editing the configuration files. To configure using the IIS Manager GUI:

1. Open Internet Information Services (IIS) Manager
2. In the Connections pane, select the site, application, or directory to be configured
3. In the Home pane, double-click Request Filtering
4. Click Edit Feature Settings... in the Actions pane
5. Under the General section, uncheck Allow double escaping

If a file name in a URL includes "+" then `allowDoubleEscaping` must be set to `true` to allow functionality.

References:

1. http://www.iis.net/ConfigReference/system.webServer/security/requestFiltering/requestLimits
2. http://learn.iis.net/page.aspx/143/use-request-filtering/

1.4.6 Disallow Unlisted File Extensions (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The `FileExtensions` Request Filter allows administrators to define specific extensions their web server(s) will allow and disallow. The property `allowUnlisted` will cover all other file extensions not explicitly allowed or denied. Often times, extensions such as `.config`, `.bat`, `.exe`, to name a few, should never be served. The `AllowExtensions` and `DenyExtensions` options are the UrlScan equivalents. It is recommended that all extensions be unallowed at the most global level possible, with only those necessary being allowed.

Rationale:

Disallowing all but the necessary file extensions can greatly reduce the attack surface of applications and servers.

Audit:

When IIS 7 rejects a request based on a file extensions filter, the error code logged is 404.7.

To manually verify the change, locate and open the `web.config` for the web site or application in which the Request Filter was set. Ensure `<fileExtensions allowUnlisted="false">`. The following `web.config` will disallow any requests for files that do not have `.asp`, `.aspx`, or `.html` as their extension:

```
<configuration>
 <system.webServer>
  <security>
   <requestFiltering>
    <fileExtensions allowUnlisted="false">
     <add fileExtension=".asp" allowed="true" />
     <add fileExtension=".aspx" allowed="true" />
     <add fileExtension=".html" allowed="true" />
    </fileExtensions>
   </requestFiltering>
  </security>
 </system.webServer>
</configuration>
```

Remediation:

The `allowUnlisted` Request Filter may be set for a server, website, or application using the IIS Manager GUI, using `AppCmd.exe` commands in a command-line window, and/or directly editing the configuration files. To configure at the server level using the IIS Manager GUI:

1. Open Internet Information Services (IIS) Manager
2. In the Connections pane, select the server
3. In the Home pane, double-click Request Filtering
4. Click Edit Feature Settings... in the Actions pane
5. Under the General section, uncheck Allow unlisted file name extensions

To set this Request Filter using an `AppCmd.exe` command, run the following command at an elevated command prompt:

```
%windir%\system32\inetsrv\appcmd set config /section:requestfiltering
/fileExtensions.allowunlisted:false
```

References:

1. http://www.iis.net/ConfigReference/system.webServer/security/requestFiltering/requestLimits
2. http://www.iis.net/learn/manage/configuring-security/configure-request-filtering-in-iis

1.4.7 Ensure Handler is not granted Write and Script/Execute (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

Handler mappings can be configured to give permissions to `Read`, `Write`, `Script`, or `Execute` depending on what the use is for - reading static content, uploading files, executing scripts, etc. It is recommended to grant a handler either Execute/Script or Write permissions, but not both.

Rationale:

By allowing both Execute/Script and Write permissions, a handler can run malicious code on the target server. Ensuring these two permissions are never together will help lower the risk of malicious code being executed on the server.

Audit:

Open the `AdministrationHost.config` file in `%windir%\system32\inetsrv\config`. Find the `<handlers>` section and verify that the `accessPolicy` attribute does not contain `Write` when `Script` or `Execute` are present. The following is an acceptable example:

```
<system.webserver>
  <handlers accessPolicy="Read, Script">
  </handlers>
</system.webserver>
```

Remediation:

The `accessPolicy` attribute in the `<handlers>` section of either the `ApplicationHost.config` (server-wide) or `web.config` (site or application) must not have `Write` present when `Script` or `Execute` are present. To resolve this issue for a Web server, the attribute in the `<handlers>` section of the `ApplicationHost.config` file for the server must manually be edited. To edit the `AdministrationHost.config` file by using Notepad, perform the following steps:

1. Open Notepad as Administrator
2. Open the `AdministrationHost.config` file in `%windir%\system32\inetsrv\config`
3. Edit the `<handlers>` section `accessPolicy` attribute so that `Write` is not present when `Script` or `Execute` are present

Note: This configuration change cannot be made by using IIS Manager.

References:

1. http://technet.microsoft.com/en-us/library/dd391910%28WS.10%29.aspx
2. http://blogs.iis.net/thomad/archive/2006/11/05/quo-vadis-accessflags.aspx

1.4.8 Ensure Configuration Attribute notListedIsapisAllowed set to false (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The `notListedIsapisAllowed` attribute is a server-level setting that is located in the `ApplicationHost.config` file in the `<isapiCgiRestriction>` element of the `<system.webServer>` section under `<security>`. This element ensures that malicious users cannot copy unauthorized ISAPI binaries to the Web server and then run them. It is recommended that `notListedIsapisAllowed` be set to `false`.

Rationale:

Restricting this attribute to `false` will help prevent potentially malicious ISAPI extensions from being run.

Audit:

Open the `applicationHost.config` file in `%windir%\system32\inetsrv\config`. Verify that the `notListedIsapisAllowed` attribute in the `<isapiCgiRestriction>` element is set to `false`:

```
<system.webServer>
  <security>
    <isapiCgiRestriction notListedIsapisAllowed="false">
    </isapiCgiRestriction>
  </security>
</system.webServer>
```

Remediation:

To use IIS Manager to set the `notListedIsapisAllowed` attribute to `false`:

1. Open IIS Manager as Administrator
2. In the Connections pane on the left, select server to be configured
3. In Features View, select ISAPI and CGI Restrictions; in the Actions pane, select Open Feature
4. In the Actions pane, select Edit Feature Settings
5. In the Edit ISAPI and CGI Restrictions Settings dialog, clear the Allow unspecified ISAPI modules check box, if checked
6. Click OK

References:

1. http://technet.microsoft.com/en-us/library/dd378846%28WS.10%29.aspx
2. http://www.iis.net/ConfigReference/system.webServer/security/isapiCgiRestriction

1.4.9 Ensure Configuration Attribute notListedCgisAllowed set to false (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The `notListedCgisAllowed` attribute is a server-level setting that is located in the `ApplicationHost.config` file in the `<isapiCgiRestriction>` element of the `<system.webServer>` section under `<security>`. This element ensures that malicious users cannot copy unauthorized CGI binaries to the Web server and then run them. It is recommended that `notListedCgisAllowed` be set to `false`.

Rationale:

Restricting this attribute to `false` will help prevent unlisted CGI extensions, including potentially malicious CGI scripts from being run.

Audit:

Browse to and open the `applicationHost.config` file and verify that the `notListedCgisAllowed` attribute in the `<isapiCgiRestriction>` element is set to `false`:

```
<system.webServer>
  <security>
    <isapiCgiRestriction notListedCgisAllowed="false">
    </isapiCgiRestriction>
  </security>
</system.webServer>
```

Remediation:

To set the `notListedCgisAllowed` attribute to false using IIS Manager:

1. Open IIS Manager as Administrator
2. In the Connections pane on the left, select the server to configure
3. In Features View, select ISAPI and CGI Restrictions; in the Actions pane, select Open Feature
4. In the Actions pane, select Edit Feature Settings
5. In the Edit ISAPI and CGI Restrictions Settings dialog, clear the Allow unspecified CGI modules check box
6. Click OK

References:

1. http://technet.microsoft.com/en-us/library/dd391919%28WS.10%29.aspx

1.4.10 Disable HTTP Trace Method (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The HTTP TRACE method returns the contents of client HTTP requests in the entity-body of the TRACE response. Attackers could leverage this behavior to access sensitive information, such as authentication data or cookies, contained in the HTTP headers of the request. One such way to mitigate this is by using the `<verbs>` element of the `<requestFiltering>` collection, which was introduced in IIS 7.0. The `<verbs>` element replaces the [AllowVerbs] and [DenyVerbs] features in IIS 6.0 UrlScan. It is recommended the HTTP TRACE method be denied.

Rationale:

Attackers may abuse HTTP TRACE functionality to gain access to information in HTTP headers such as cookies and authentication data. This risk can be mitigated by not allowing the TRACE verb.

Audit:

IIS 8 will return an HTTP 404.6 error to the client when Request Filtering blocks an HTTP request because of a denied HTTP verb. To manually verify the change, browse to the `web.config` file for which the change was made and verify the below configuration:

```
<configuration>
  <system.webServer>
    <security>
       <requestFiltering>
          <verbs>
            <add verb="TRACE" allowed="false" />
          </verbs>
        </requestFiltering>
     </security>
  </system.webServer>
</configuration>
```

Remediation:

1. Open Internet Information Services (IIS) Manager
2. In the Connections pane, select the site, application, or directory to be configured
3. In the Home pane, double-click Request Filtering
4. In the Request Filtering pane, click the HTTP verbs tab, and then click Deny Verb... in the Actions pane
5. In the Deny Verb dialog box, enter the TRACE, and then click OK

References:

1. http://www.kb.cert.org/vuls/id/867593
2. http://www.iis.net/ConfigReference/system.webServer/security/requestFiltering/verbs

1.5 IIS Logging Recommendations

This section contains recommendations regarding IIS logging that have not been covered in the Basic Configurations section.

1.5.1 Move Default IIS Web Log Location (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

IIS will log relatively detailed information on every request. These logs are usually the first item looked at in a security response, and can be the most valuable. Malicious users are

aware of this, and will often try to remove evidence of their activities. It is therefore recommended that the default location for IIS log files be changed to a restricted, non-system drive.

Rationale:

Moving IIS logging to a restricted, non-system drive will help mitigate the risk of logs being maliciously altered, removed, or lost in the event of system drive failure(s).

Audit:

To verify web logs are being logged to the new location, open Windows Explorer and browse to the path that was defined. Depending on how the logging was configured, there will be either:

1. A folder containing .log files or
2. .log files in the root of the specified directory

Remediation:

Moving the default log location can be easily accomplished using the Logging feature in the IIS Management UI or `AppCmd.exe`. To change to D:\LogFiles using `AppCmd.exe`:

```
%windir%\system32\inetsrv\appcmd set config -section:sites -
siteDefaults.logfile.directory:"D:\LogFiles"
```

Moving log file stores to a non-system drive or partition separate from where web applications run and/or content is served is preferred. Additionally, folder-level NTFS permissions should be set as restrictive as possible; Administrators and SYSTEM are typically the only principals requiring access.

Note: While standard IIS logs can be moved and edited using IIS Manager, additional management tool add-ons are required in order to manage logs generated by other IIS features, such as Request Filtering and IIS Advanced Logging. These add-ons can be obtained using the Web Platform Installer or from Microsoft's site. The HTTPErr logging location can be changed by adding a registry key

References:

1. http://www.iis.net/learn/extensions/advanced-logging-module/advanced-logging-for-iis-custom-logging
2. http://support.microsoft.com/kb/820729#1

1.5.2 Enable Advanced IIS Logging (Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

IIS Advanced Logging is a module which provides flexibility in logging requests and client data. It provides controls that allow businesses to specify what fields are important, easily add additional fields, and provide policies pertaining to log file rollover and Request Filtering. HTTP request/response headers, server variables, and client-side fields can be easily logged with minor configuration in the IIS management console. It is recommended that Advanced Logging be enabled, and the fields which could be of value to the type of business or application in the event of a security incident, be identified and logged.

Rationale:

Many of the fields available in Advanced Logging many can provide extensive, real-time data and details not otherwise obtainable. Developers and security professionals can use this information to identify and remediate application vulnerabilities/attack patterns.

Audit:

Browse to the location of the Advanced Logs and verify .log files are being generated. Note that logs will be written to disk after a non-determined period of time. They can be written into their specified directory immediately if, in the Log Definition, the Publish real-time events and Write to disk options are selected.

Remediation:

IIS Advanced Logging can be configured for servers, Web sites, and directories in IIS Manager. To enable Advanced Logging using the UI:

1. Open Internet Information Services (IIS) Manager
2. Click the server in the Connections pane
3. Double-click the Advanced Logging icon on the Home page
4. Click Enable Advanced Logging in the Actions pane

The fields that will be logged need to be configured using the Edit Logging Fields action. As with IIS's standard log files, their location should be changed.

Note: There may be performance considerations depending on the extent of the configuration.

References:

1. http://learn.iis.net/page.aspx/579/advanced-logging-for-iis-70---custom-logging#open
2. http://technet.microsoft.com/en-us/library/cc732826%28WS.10%29.aspx

1.5.3 ETW Logging (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.5

Description:

IIS 8.5 introduces a new logging method. Administrators can now send logging information to Event Tracing for Windows (ETW)

Rationale:

IIS flushes log information to disk, therefore prior to IIS 8.5, administrators do not have access to real-time logging information. Text-based log files can also be difficult and time consuming to process. By enabling ETW, administrators have access to use standard query tools for viewing real-time logging information.

Audit:

Using Message Analyzer, configure the query for Microsoft-Windows-IIS-Logging. Verify you see live logging data by accessing the website.

Remediation:

To configure ETW logging:

1. Open **IIS Manager**

2. Select the server or site to enable ETW

3. Select **Logging**.

4. Ensure Log file format is **W3C**.

5. Select Both log file and ETW event

6. Save your settings.

References:

1. http://www.iis.net/learn/get-started/whats-new-in-iis-85/logging-to-etw-in-iis-85
2. http://blogs.technet.com/b/erezs_iis_blog/archive/2013/07/15/hook-me-up.aspx

1.6 FTP Requests

This section contains a crucial configuration setting for running file transfer protocol (FTP).

1.6.1 Encrypt FTP Requests (Not Scored)

Profile Applicability:

- Level 1 - IIS 8.0

Description:

The new FTP Publishing Service for IIS 7.0 supports adding an SSL certificate to an FTP site. Using an SSL certificate with an FTP site is also known as FTP-S or FTP over Secure Socket Layers (SSL). FTP-S is an RFC standard (RFC 4217) where an SSL certificate is added to an FTP site and thereby making it possible to perform secure file transfers.

Rationale:

By using SSL, the FTP transmission is encrypted and secured from point to point and all FTP traffic as well as credentials are thereby guarded against interception.

Audit:

The FTP site will now require the use of FTP-S; test this by attempting to use an FTP client which either does not support FTP-S or is not configured to use FTP-S. If setup was successful, the request will fail. Conversely, open a command prompt from the server and type `ftp localhost`. After entering credentials, the server should return an Access is Denied message.

Remediation:

To secure an existing FTP site using a SSL Certificate, a certificate must first be installed on the system. Production systems should always use a third party certificate from a trusted root, such as VeriSign. Once that certificate is installed for use in IIS, follow the steps below to configure the FTP site for SSL:

1. Open IIS Manager, select the FTP server and choose FTP SSL Settings in the Features View pane
2. Under the SSL Certificate dropdown, choose the SSL certificate to be configured for use
3. In the SSL Policy section, click the radio button next to Require SSL connections; it is important to require SSL, because allow SSL still permits non-SSL FTP
4. Click Apply in the Actions pane

References:

1. http://www.windowsnetworking.com/articles_tutorials/IIS-FTP-Publishing-Service-Part3.html
2. http://learn.iis.net/page.aspx/304/using-ftp-over-ssl/#03

Appendix: Change History

Date	Version	Changes for this version
01-06-2014	1.0.0	Initial release.