ERNW
providing security.

Security Hardening

Microsoft IIS 7.5

Version:	1.0.0
Date:	6/26/2015
Classification:	Public
Author(s):	Dominik Phillips, Friedwart Kuhn

Table of Content

1 Introduction

Internet Information Services (IIS) contains several components that perform important functions for the application and Web server roles in Windows Server. As it is designed to be used in an enterprise environment, the security of this system must be kept at a high level.

By default IIS implements a lot of basic security measures, but are these the relevant ones to protect your business?

In order to answer this question for one of our customers, we have compiled the most relevant security settings in an IIS 7.5 Hardening Guide for you. In this guide we define a baseline security level, which is to be used for so called "crash and burn systems" (systems with non-critical data, systems whose availability have no business relevant impact) and a security level high, which includes all other systems. The mitigations in the baseline section are non-critical and therefore no further test are necessary. The mitigation in the section high, are critical in terms of availability and need to be tested extensively. The system owner must decide, which security level is the right one for their system, and which mitigation from section high are mandatory for their system.

The IIS 7.5 Hardening Guide includes configuration examples and all necessary commands for each mitigation.

2 OVERVIEW

This document defines security requirements for IIS 7.5. The document regulates the security level by defining requirements in two security levels – **baseline** and **high** security.

Hardening requirements in section 3 may have different attributes such as:

✋ Requirement can be implemented manually

🖧 Requirement can be implemented by organizational means (e.g. process, agreement, etc.)

Requirement descriptions in section 3 may contain the following attributes:

⚙ Guidance on how to implement the requirement

⚠ Implementing this requirement may impact functionality or change the behavior of the service.

2.1 How to Use this Document

2.2 Document Scope

In Scope

The document scope is a basis installation of Microsoft IIS 7.5 Web Server. It specifies security options that are compliant with the corporate security requirements for application servers.

Out of Scope

Additional Microsoft server roles and features such as AD, File Server, DNS, etc. are not in scope of this document.

3 Security Requirements

This section contains all hardening requirements divided in two levels - section 3.1 and 3.2. The "High" hardening requirements are in addition, which implies that to implement level "High" you need to implement both requirements for Baseline and High.

3.1 Security Level – Baseline

Ensure security of standard and default accounts

BR001: Built-In user and group accounts in IIS 7.5	✋ 🖧

All predefined user accounts and groups in IIS 7.5 are build-in accounts and require no further security configuration.

Ensure password security

BR002: Implement secure password policy	✋ 🖧

A password policy must be defined for all users and should be set as follows:

- Minimum password length of 8 characters
- Password must consist of at least one character of each character group (letters, capital letters, numbers, special characters)
- Must not contain any default passwords
- Must consist of at least 6 different characters
- Should have a maximum age of 180 days
- Username must not be part of password
- At least the 5 previous passwords must not be (nearly) equal

⚠ *This requirement should already be technically implemented on OS level*

Ensure usage of latest version (patch management)

BR003: Implement appropriate patch management for the IIS web server role	🖧

Ensure that the most current patches, updates and hot-fixes are implemented in a timely manner:

- All Windows OS components shall be up-to-date.
- All Microsoft IIS Server components shall be up-to-date.
- A process shall be in place to regularly update all software components.
- Security corrections shall be implemented in a timely manner.
- Very High and High priority updates and patches are to be deployed after a maximum of 4 weeks after the update release.

- Medium and Low priority updates and patches are to be deployed after a maximum of 8 weeks after the update release.
- Security updates and patches shall be deployed and monitored centrally.

Central deployment tools such as WSUS and BladeLogic shall be employed for central management.

Use service accounts with minimal privileges

BR004: Ensure that each application pool run under a unique identity	✋

Application Pool Identities are „Managed Local Accounts” which allows you to run each Application Pool under a unique authority.

Verify that each Application Pool has been set to run under an unique identity:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Open the Application pools node
- Right click the Application Pool and select Advanced Settings
- Locate the Identity option and ensure that ApplicationPoolIdentity is set

⚠ *The above-mentioned setting must not be verified on freshly installed IIS 7.5 web servers because it already set by default.*

Use secure authentication

BR005: Ensure access to sensitive site features is restricted to authenticated principals only	✋

IIS 7.5 supports two authentication method groups; challenge-based and login redirection-based authentication.

Sites containing sensitive information, confidential data, or non-public web services must be configured with a credential-based authentication mechanism.

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Navigate to the level with sensitive content
- Select Authentication
- On the authentication page, make sure an authentication method is enabled
- If necessary, select the desired authentication method

⚠ *By default the following authentication procedures are available:*

- *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*

BR006: Configure SSL when using “Forms Authentication” ✋

The login redirection-based authentication method “Forms Authentication” can pass user name and password across the network in clear text. Therefore traffic between client and server must be encrypted using SSL.

Verify the traffic between client and server is encrypted using CMD:

⚙ Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ `“type web.config | findstr “forms””`

- Verify that the tag looks like this:

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

Verify the traffic between client and server is encrypted by using IIS Manager:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Navigate to the appropriate tier
- In Features View, select Authentication
- Select Forms Authentication
- Select Edit
- Verify the Requires SSL checkbox and set it if not already activated

BR007: Configure SSL when using “Basic Authentication” ✋

The login redirection-based authentication method “Basic Authentication” will pass user name and password across the network in clear text. Therefore traffic between client and server must be encrypted using SSL.

Configure Basic Authentication with SSL:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Select the site to be configured
- In the Actions pane, select bindings
- If an HTTPS binding is available, see below "Require SSL"
- If no HTTPS binding is visible, perform the following steps “Add an HTTPS binding”

Add an HTTPS binding:

- In the Actions pane, select bindings
- Add an new binding
- Under Type, select the “https” protocol
- Under SSL certificate, select an SSL certificate

Require SSL:

- In Features View, select SSL Settings
- On the SSL Settings page, select Require SSL, and Require 128-bit SSL
- In the Actions pane, “Apply” settings

Ensure secure interconnectivity

BR008: Ensure the web server has a valid X.509 certificate	⚙

If communication leaves the host system, it must be appropriately secured (e.g. by SSL).

To configure an SSL certificate for IIS perform the following:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager
Create an Certificate Request:

- In the connections pane select the Web Server and then select Server Certificates
- In the Action pane click “Create Certificate Request” and then follow the instructions
- Sign the Certificate Request by a CA (see Notes below)

Complete the Certificate Request

- In the connections pane select the Web Server and then select Server Certificates
- In the Action pane click “Complete Certificate Request” and then follow the instructions

⚠ *The server certificate should meet the following requirements:*

- *Certificate issuer (CA) must be a trusted authority.*

   - *For internal scenarios use the internal Certificate Authority (must NOT be used as client certificates).*
   - *For Internet-based scenarios where also third parties would consume the service and do not trust the internal CA, use well-known and trusted CAs such as TrustCenter.*
- *Certificate Key should be 2048 bits or higher (1024bit keys are for the time being also considered appropriate, although 2048 should be preferred)*
- *The certificate should be signed using a secure Hash algorithm (e.g. SHA (Secure Hash Algorithm))*
- *Validity period of certificates:*

For external Internet scenarios – max. 1 year

For internal scenarios – max. 2 years

Schedule timely refresh of SSL certificates before these are expired

- *CN name of the certificate must be identical to the server DNS name under which the service is known to the client.*

Use a 'minimal principle' authorization concept

BR009: Usage of personalized administrative accounts	

To ensure traceability of actions done by a specific user, each administrative user must have a personalized account with only the absolute necessary rights.

⚠ *The above-mentioned administrative user accounts are accounts on the OS level and are not configured via IIS.*

BR010: Use dedicated database account	🖧

If IIS applications running on the same host make use of databases it must be ensured that each application is using its own database account and if appropriate, a separate database instance for each application is maintained. The database accounts must not be able to view or alter data in other databases.

⚠ *The above-mentioned statement is a general best practices database separation of duties procedure and is configured in the database.*

Implement baseline application server hardening

BR011: Ensure web content is not on the OS system partition/drive	✋

Web content is mapped via "Virtual Directories" to physical locations on the disk. Do not use the default `<\inetpub\wwwroot>` directory.

- Isolating the web content from the operating system must be implemented by deploying it on different file system partitions or physical hard drives. Whenever possible, you should separated it on a physical hard drive.

BR012: Directory Browsing	✋

Directory browsing allows the contents of a directory to be displayed over a request from a web client.

- If you don't need Directory Browsing, Directory Browsing must be disabled.

Verify that Directory Browsing has been disabled:

Server level:

⚙ Go to: Start → Run → start „cmd" → `"%systemroot%\system32\inetsrv\appcmd list config /section:directoryBrowse"`

Application level:

⚙ Go to: Start → Run → start „cmd" → `"%systemroot%\system32\inetsrv\appcmd list config "Site Name" /section:directoryBrowse"`

- If Directory Browsing is disabled, the following should be displayed

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

To disable Directory Browsing perform the following:

Server level:

⚙ Go to: Start → Run → start „cmd" → `"%systemroot%\system32\inetsrv\appcmd set config /section:directoryBrowse /enabled:false /showFlags: ["Date, Time, Size, Extension, LongDate, None"]"`

Application level:

⚙ Go to: Start → Run → start „cmd" → `"%systemroot%\system32\inetsrv\appcmd list config "Site Name" /section:directoryBrowse /enabled:false /showFlags: ["Date, Time, Size, Extension, LongDate, None"]"`

⚠ If directory browsing is enabled for a directory in IIS 7.5, clients receives the content of a directory when the following two conditions are met:

- No specific file is requested in the URL (example: "*http://www.testurl.net/*")
- The Default Documents feature is disabled, or if it is enabled and the IIS Server is unable to locate a file in the directory that matches a name specified in the Default Document list.

BR013: Configure host headers on all sites	✋

Host headers provide ability to host multiple websites on the same IP address and port. In addition, it is an easy to implement and effective measure against e.g. IP based scans and DNS rebinding attacks.

- Host headers must be configured for all sites, check and set values as follows:

Identify sites that are not configured to require host headers:

⚙ Go to: Start → Run → start „cmd" → `"%systemroot%\system32\inetsrv\appcmd list sites"`

All sites will be listed as such:

```
SITE "Default Web Site" (id:1,bindings:http/IP:PORT:HOST,state:Started)
```

Ensure that for all sites the IP:PORT:HOST binding contains a host name.

Configure host headers in IIS Manager:

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the connections pane expand the Sites node and select your Web Site
- In the Action pane click Bindings
- In the Site Bindings dialog box, select the bindings for which host headers are going to be configured
- Under host name, enter the sites FQDN

BR014: Ensure unique application pools for sites	✋

An application pool defines a group of one or more worker processes, which serve requests to one or more applications that are assigned to that application pool.

- All Sites should be running under a unique, dedicated Application Pools.

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:

⚙ Go to: Start → Run → start „cmd" → "`%systemroot%\system32\inetsrv\appcmd list app`

⚠ *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. Finally, by isolating applications it helps mitigate the potential risk of one application being allowed access to the resources of another application.*

BR015: Configure anonymous user identity to use application pool identity	✋

The IIs 7.5 must be configured to use the Application Pool Identity for anonymous user accounts authentication.

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- IIS Manager GUI: navigate to the desired server, site or application
- In Features View select the Authentication icon
- Select the Anonymous Authentication option and in the Action pane select Edit
- Select the Application Pool Identity in the model window

⚠ *The above-mentioned setting must not be verified on freshly installed IIS 7.5 Web Servers. The above-described value is already set by default.*

BR016: Configure “Forms Authentication” to use cookie session management ✋

Forms Authentication can be configured to maintain the site visitor's session identifier in either a URI or cookie.

- If the web application uses Forms Authentication it must be configured to use cookies.

Verify the traffic between client and server is encrypted using CMD:

⚙ Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ `“type web.config | findstr “forms””`

- Verify that the tag looks like this:

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

Verify the traffic between client and server is encrypted using IIS Manager:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Navigate to the appropriate tier
- In Features View, select Authentication
- Select Forms Authentication
- Select Edit
- In the Cookie settings section, select Mode and from the dropdown select “Use cookies”

BR017: Configure cookie protection mode for “Forms Authentication” ✋

The cookie protection mode defines the protection Forms Authentication cookies will be given within a configured application.

- If the web application uses Forms Authentication. Cookie protection mode must always encrypt and validate Forms Authentication cookies.

Verify the traffic between client and server is encrypted using CMD:

⚙ Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ `“type web.config | findstr “forms””`

- Verify that the tag looks like this:

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

Verify the traffic between client and server is encrypted using IIS Manager:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Navigate to the appropriate tier
- Select Authentication
- Select Forms Authentication
- Select Edit
- In the Cookie settings section, select Protection mode and from the drop-down select "Encryption and validation"

⚠ *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.*

The following four cookie protection modes can be defined:

- *Encryption and validation:*
  Specifies that the application use both data validation and encryption to help protect the cookie.
- *None:*
  Specifies that both encryption and validation are disabled.
- *Encryption:*
  Specifies that using Triple-DES or DES encrypts the cookie.
- *Validation:*
  Specifies that a validation scheme verifies that the contents of an encrypted cookie have not been changed.

BR018: Set deployment method to retail	✋

The `<deployment retail>` switch is used to help applications to run with the best possible performance and least possible security information leakages.

- The deployment method on any production server must be set to `<retail="true">`.

Verify that Deployment Method to Retail is set to:

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:
   `%windir%\Microsoft.NET\Framework64\<framework_version>\CONFIG`

BR019:.Net: Ensure custom error messages are not Off	✋

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.

- On productive systems the setting must be set to one of the following values: On or RemoteOnly

Use the CMD shell to verify this setting:

⚙ Go to: Start → Run → start „cmd" → navigate to the configured site / application / content location→ "`type web.config | findstr "customErrors"`"

```
<configuration>
       <system.web>
              <customErrors mode="RemoteOnly" /> or <customErrors mode="On"
       </system.web>
</configuration>
```

Use the UI to make this change:

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Navigate desired server, site, or application
- In Features View, select .NET Error Pages
- In the Actions Pane, select Edit Feature Settings
- In modal dialog, choose On or Remote Only for Mode settings

⚠ *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.

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.*

BR020: Ensure failed request tracing is not activated	✋

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. In order to use tracing, it must be installed as a role service under the Health and Diagnostics section of the Web Server role.

- This setting must be configured on productive systems.

Use the CMD shell to configure or verify this setting:

Verify

e.g. ⚙ Go to: Start → Run → start „cmd" → `"%windir%\system32\inetsrv\appcmd.exe list config "default web site" | findstr "traceFailedRequestsLogging""`

Config

e.g. ⚙ Go to: Start → Run → start „cmd" → `"%windir%\system32\inetsrv\appcmd configure trace "Default Web Site" /disablesite"`

Verify Failed Request Tracing is turned off by using the IIS Manager GUI:

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the Connections pane, select the server connection, site, application, or directory on which failed request tracing will be configured
- In the Actions pane, select Failed Request Tracing...

- In the Edit Web Site Failed Request Tracing Settings dialog box, verify that the Enable check box is not checked

⚠ *This is a defense in depth recommendation due to the* `<deployment retail="true" />` *in the* `machine.config` *file overriding any settings for Failed Request Tracing to be left on. It is recommended that Failed Request Tracing still be turned off.*

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

BR021: Ensure cookies are set with HttpOnly attribute ✋

This setting should be deactivated for productive systems.

When cookies are set with the `HttpOnly` flag, they cannot be accessed by client-side scripting, running in the user's browser.

- On productive system the `httpOnlyCookies` attribute must be set to true.

Use the CMD shell to configure or verify this setting:

Verify

⚙ Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ “`type web.config | findstr "httpCookies"`”

Config

⚙ Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ “notepad web.config”

- Add the `<httpCookies httpOnlyCookies="true" />` tag within `<system.web>`.

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

⚠ *The above-mentioned setting must not be verified on freshly installed IIS 7.5 Web Servers. The above-described value is already set by default.*

BR022: Configure global .NET trust level ✋

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.

Use the CMD shell to configure or verify this setting:

Verity

⚙ Go to: Start → Run → start „cmd“ → `“%windir%\system32\inetsrv\appcmd.exe list config "default web site" /section:trust”`

Config

⚙ Go to: Start → Run → start „cmd“ → `“%systemroot%\system32\inetsrv\appcmd set config /commit:WEBROOT /section:trust /level:Medium”`

Verify the global .NET Trust Level using IIS Manager:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- *Navigate to the level that was configured,* the WEBROOT, or server
- In the features view, select .NET Trust Levels
- On the .NET Trust Levels page, verify that Medium (web_mediumtrust.config) is selected in the Trust Level dropdown

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

BR023: Hide detailed errors from displaying remotely	✋

To prevent unauthorized users from viewing detailed error informations, 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.

- On productive systems detailed errors must be prevented from displaying remotely.

Use the CMD shell to configure or verify this setting:

<u>Verity</u>

- ⚙Go to: Start → Run → start „cmd“ → `“%windir%\system32\inetsrv\appcmd.exe list config "default web site" /section:httpErrors”`

AND

- ⚙Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ `“type web.config | findstr "httpErrors"”`

Verify the errorMode is set to `DetailedLocalOnly` or `Custom`:

- `<httpErrors errorMode="[DetailedLocalOnly or Custom]">`

<u>Config</u>

⚙ Go to: Start → Run → start „cmd“ → `“%systemroot%\system32\inetsrv\appcmd set config “default web site” /section:httpErrors /errorMode:[Custom or DetailedLocalOnly]”`

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

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- *Navigate to the level that was configured,* the WEBROOT, or server
- In Features View, select Error Pages
- In the Actions pane, select Edit Feature Settings
- 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

BR024: Encrypt FTP requests	✋

The FTP Publishing Service supports adding an SSL certificate to an FTP site. 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.

To secure an existing FTP site using a SSL Certificate, a certificate must first be installed on the system. Once that certificate is installed for use in IIS, follow the steps below to configure the FTP site for SSL:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Select the server
- Select the FTP server and choose FTP SSL Settings in the Features View pane
- Under the SSL Certificate dropdown, choose the SSL certificate to be configured for use
- 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

Implement backup and disaster recovery procedures

BR025: Backup procedure

Use the backup process to allow a restore of the complete system (including databases and configurations). If the backup process cannot be used, implement a backup process in compliance with the corporate Backup Infrastructure directive. The backups must be stored encrypted. The transfer of the backups must be done in a secure way.

The following should be backed up:

- Everything that is stored in `%systemdrive%\inetpub\history`
    - IIS global configuration (*applicationHost.config)*
    - Admin Tool's configuration *(administration.config)*
    - All `*.xml` files that are stored under schema
- metabase.xml and mbschema.xml are stored in *`%SystemRoot%\system32\inetsrv\`*
    - Metabase data that is still used by some IIS services including SMTP and FTP, that have not been migrated to the new configuration system.
- Local stored application data

⚠ *How to backup/restore IIS configuration http://blogs.iis.net/bills/archive/2008/03/24/how-to-backup-restore-iis7-configuration.aspx*

Secure integration of external web-services

BR026: Secure integration of external web services

To integrate an internal IIS server with an external (Internet) Web-Service, the following measures must be applied:

- Only connections from the Corporate Network to the Internet Web-Service are allowed. Inbound traffic is not allowed.
- Use the official corporate Internet proxies only.

- Do not use your private Internet Web-Service (e.g. Twitter, Facebook) account for development activities.
- Ensure your content/data is polite, respectful, and honest.
- Do not use anonymous Internet Web-Service accounts. Always make clear who you are (identify as your organization).
- Use passwords for the Internet Web-Service accounts that follow the corporate security policy.
- Use encrypted protocols if technically possible (e.g. HTTPS).
- Do not violate Terms & Conditions of the Internet Web-Service provider.
- Do not upload or download data classified as "internal" or higher to/from the Internet Web-Service.
- Document your Internet Web-Service usage and configuration in your project/landscape documentation.
- In case of mass data processing: create a Security Concept and get a security approval by your CERT (Computer Emergency Response Team).

3.2 Security Level – High

Separate end user access from administration/configuration access	
HR001: Isolate administrative web server access	✋ 🖧
End user access to a particular application should be isolated from administrative access. Administrative access should be done via a separate network port using an encrypted protocol. The administrative network must be segregated by means of firewall rules and/or network ACLs in order to ensure that only authorized administrative personnel has access to the OS administrative interfaces. Direct access to RDP, RPC, etc. ports used for administration must be possible only over an isolated network which is accessed over dedicated jump hosts.	
Reduce attack surface area	
HR002: Reducing the attack surface by web server features	✋ 🖧
IIS is designed with a modular architecture and a minimum of module and features installed per default. You can choose from 40 modules to customize your installation for the needs of your particular web server. • Only those modules and features must be installed that are needed for the operation of your environment	
Use secure communication	
HR003: Securing communication with SSL	🖧
To secure the administrative interface and every web application communication, the usage of SSL must be configured. For this purpose create a trustworthy certificate to avoid certificate-warning messages and offer the end user a way to verify a trustworthy connection. Refer to chapter BR014 for further details	

HR004: Use only strong encryption protocols ✋

SSL-based services should not offer the possibility to utilize weak encryption protocols or ciphers.

⚙Verify that the protocols are configured as follows:

- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\PCT 1.0\Server\Enabled
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL 2.0\Server\Disabled
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL 3.0\Server\Enabled
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0\Server\Enabled
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.1\Server\DisabledByDefault
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server\DisabledByDefault

HR005: Disable weak cipher suites ✋

SSL-based services should not offer the possibility to utilize weak protocols or ciphers.

⚙Verify that the ciphers are configured as follows:

- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\DES 56/56
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\NULL
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC2 40/128
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4 40/128
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4 56/128
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4 64/128
  - All this cipher are disabled by default on Windows Server 2008 R2
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4 128/128
  - RC4 128/128 is enabled by default on Windows Server 2008 R2
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\Triple DES 168/168
  - DES 168/168 is enabled by default on Windows Server 2008 R2
- HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AES 256/256
  - AES 256/256 is enabled by default on Windows Server 2008 R2

Implement multiple lines of defense

HR006: Make use of the Enhanced Mitigation Experience Toolkit ✋ 🖧

The Enhanced Mitigation Experience Toolkit (EMET) is a utility that helps prevent vulnerabilities in software from being successfully exploited. EMET achieves this by using security mitigation technologies. These technologies function as special protections and obstacles that an exploit author must defeat to exploit software vulnerabilities.

- To increase security of the IIS Web Server, EMET should be used.

- To download EMET, visit the following Microsoft website: *http://www.microsoft.com/en-us/download/details.aspx?id=29851*

⚠ *This can break functionality of some applications and might increase the effort on debugging application errors. Advanced settings need to be tested extensively before they go live on productive systems.*

- *https://www.ernw.de/download/emet/emet_walkthrough_v.1.0_signed.pdf*

HR007: Dynamic IP Restrictions ✋ 🖧

The Dynamic IP Restrictions Extension helps mitigate or block Denial of Service Attacks or cracking of passwords through brute force by temporarily blocking IP addresses of HTTP client who follow a pattern that could be conducive to one of such attacks.

- For applications, which are only available from the corporate network, dynamic IP restrictions should be implemented.

- To download Dynamic IP Restrictions Extension, visit the following website: *http://www.iis.net/downloads/microsoft/dynamic-ip-restrictions*

Monitor security configuration

HR008: Enabling configuration auditing ✋ ⚙

IIS configuration auditing let you monitor the changes that are done to the IIS configuration. It generates Microsoft Event messages, which display the configuration element, which was changed, user who initiated the change, and the original and the new value of the element.

- Configuration auditing must be enabled

To enable the configuration auditing feature, follow the below steps:

Open Event Viewer (Administrative Tools –> Event Viewer)

Expand the "Application and Service Logs"

Expand "Microsoft", and expand "Windows"

Expand "IIS-Configuration", and right click on "Operational", and choose "Enable Log"

⚠ *Define which security configurations must be monitored.*

Configure secure session handling

HR009: Configure use cookies mode for session state	✋

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 URL.

- This setting must be configured on productive systems to `UseCookies`.

Use the CMD shell to configure or verify this setting:

Verify

⚙ Go to: Start → Run → start „cmd" → `"%windir%\system32\inetsrv\appcmd.exe list config "default web site" /section:system.web/sessionState"`

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

Config

⚙ Go to: Start → Run → start „cmd" → `"%windir%\system32\inetsrv\appcmd set config /commit:WEBROOT /section:sessionState /cookieless:UseCookies /cookieName:ASP.NET_SessionID /timeout:20"`

Perform the following to set the cookieless attribute in the IIS Manager GUI:

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Navigate desired server, site, or application

- In Features View, select Session State
- In the Cookie Settings section, choose Use Cookies from the Mode dropdown
- In the Actions Pane, Apply settings

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

HR010: Configure maxAllowedContentLength request filter	✋

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.

- The overall size of requests should be restricted to a maximum value appropriate for the server, site, or application.

Use the CMD shell to configure or verify this setting:

Verity

- ⚙Go to: Start → Run → start „cmd" → "`%windir%\system32\inetsrv\appcmd.exe list config "default web site" /section:requestFiltering`"

Verify the `maxAllowedContentLength` is set to `30000000`:

- `<requestLimits maxAllowedContentLength="30000000">`

Config

- ⚙Go to: Start → Run → start „cmd" → navigate to the configured site / application / content location→ "`notepad web.config`

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

OR

- ⚙ Go to: Start → Run → start „cmd" → "`%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /requestLimits.maxAllowedContentLength:30000000`"

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

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the Connections pane, select the server, site, application, or directory
- Select Request Filtering

- In the Actions pane, select Edit Feature Settings
- 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)

HR011: Configure maxURL request filter	✋

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.

- A limit should be put on the length of URL.

Use the CMD shell to configure or verify this setting:

Verity

- ⚙Go to: Start → Run → start „cmd" → "`%windir%\system32\inetsrv\appcmd.exe list config "default web site" /section:requestFiltering`"

Verify the `maxUrl` is set to `2048`:

- `<requestLimits maxUrl="2048">`

Config

- ⚙Go to: Start → Run → start „cmd" → navigate to the configured site / application / content location→ "`notepad web.config`

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

OR

- ⚙ Go to: Start → Run → start „cmd" → "`%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /requestLimits.maxUrl:2048`"

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

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the Connections pane, select the server, site, application, or directory
- Select Request Filtering
- In the Actions pane, select Edit Feature Settings
- Under the Request Limits section, key the maximum URL length in bytes that has been tested with web applications

HR012: Configure MaxQueryString request filter	✋

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.

- Values always should be established to limit the amount of data will can be accepted in the query string.

Use the CMD shell to configure or verify this setting:

Verity

- ⚙Go to: Start → Run → start „cmd" → "`%windir%\system32\inetsrv\appcmd.exe list config "default web site" /section:requestFiltering`"

Verify the `maxUrl` is set to `1024`:

- `<requestLimits maxQueryString="1024">`

Config

- ⚙Go to: Start → Run → start „cmd" → navigate to the configured site / application / content location→ "`notepad web.config`

```
<system.webServer>
      <security>
            <requestFiltering>
                  <requestLimits maxQueryString="1024"/>
            </requestFiltering>
      </security>
</system.webServer>
```

OR

- ⚙ Go to: Start → Run → start „cmd" → "`%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /requestLimits.maxQueryString="1024`"

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

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the Connections pane, select the server, site, application, or directory
- Select Request Filtering
- In the Actions pane, select Edit Feature Settings
- Under the Request Limits section, key in a safe upper bound in the Maximum query string (Bytes) textbox

HR013: Configure the ASP session state attributes	✋

The `<session>` attribute of the Active Server Page `<asp>` element specifies the session state setting. The session state attribute defines how the IIS Web Server stores information about each unique client session which is configure to yours ASP.

Use the CMD shell to configure or verify this setting:

Verify

web.config (application site)

- ⚙ Go to: Start → Run → start „cmd" →„%systemroot%/system32/inetsrv/appcmd list config „default web site" /section:system.webServer/asp"

machine.config (server site)

- ⚙ Go to: Start → Run → start „cmd" →„%systemroot%/system32/inetsrv/appcmd list config /section:system.webServer/asp"

Ensure that the following attributes <allowSessionState>, <max> and <timeout> are set as followed:

```
<location path="Default Web Site">
   <system.webServer>
      <asp>
         <session allowSessionState="true" max="1000" timeout="00:10:00" />
      </asp>
   </system.webServer>
</location>
```

Config

web.config (application site)

- <allowSessionState>
- ⚙ Go to: Start → Run → start „cmd" → "%systemroot%/sytem32/inetsrv/appcmd set config /section:system.webServer/asp /session.allowSessionState:[true or false]"
- <max>
- ⚙ Go to: Start → Run → start „cmd" → "%systemroot%/system32/inetsrv/appcmd set config /section:system.webServer/asp /session.max:1000"
- <timeout>
- ⚙ Go to: Start → Run → start „cmd" → "%systemroot%/system32/inetsrv/appcmd set config /section:system.webServer/asp /session.timeout:"00:10:00""

machine.config (Server site)

- <allowSessionState>
- ⚙ Go to: Start → Run → start „cmd" →„%systemroot%/sytem32/inetsrv/appcmd set config /section:system.webServer/asp /session.allowSessionState:[true or false]"
- <max>
- ⚙ Go to: Start → Run „cmd" → „%systemroot%/sytem32/inetsrv/appcmd set config

/section:system.webServer/asp /session.max:“1000““

- <timeout>
- ⚙ Go to: Start → Run „cmd“ → „%systemroot%/sytem32/inetsrv/appcmd set config /section:system.webServer/asp /session.timeout:“00:10:00““

Use IIS Manager to configure or verify the ASP Session State:

⚙ Go to: Start → Run „control“ → navigate to Administrative Tools → Internet Information Services (IIS) Manager

- <timeout>
- Navigate to the Server, Web site ore Web Application you want to configure
- Select the ASP icon
- In the Asp pane, expand the Session Properties and configure the following settings
    - <allowSessionState>: [true or false]
    - <max>: [1000]
    - <timeout>: [00:10:00]

Enable logging for critical application events

HR014: Enable advanced IIS logging	✋

IIS Advanced Logging is a module, which provides flexibility in logging requests and client data.

- Advanced Logging must 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.

IIS Advanced Logging can be configured for servers, Web sites, and directories in IIS Manager.

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Select the server
- Select Advanced Logging icon on the Home page
- 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.

⚠ *There may be performance considerations depending on the extent of the configuration.*

HR015: Move default IIS web log location	✋

IIS logs relatively detailed information on every request.

- The default location for IIS log files must be changed to a restricted, non-system drive.

Use the CMD shell to configure or verify this setting:

- Go to: Start → Run → start „cmd" → "`%windir%\system32\inetsrv\appcmd set config /section:sites /siteDefaults.logfile.directory:"D:\LogFiles"`"

Use IIS Manager to set the new log file location:

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Select the server
- In the Home pane, select Logging
- In the Directory dialog box, enter the log file location "e.g. D:\LogFiles"

⚠ *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:*

- *A folder containing .log files or log files in the root of the specified directory*

Create and document an authorization concept

HR016: Create and document an authorization concept	🖧

Create and document a formalized authorization concept based on roles and groups. The authorization concept can be documented as part of the security concept document - see Quick Link /go/security-concepts.

Implement intense application server hardening

HR017: Ensure double encoded requests will be rejected	✋

When the double-encoded requests filter is enabled, IIS 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.

- Double-encoded requests must be rejected.

For operational reasons this setting must be enabled.

Use the CMD shell to configure or verify this setting:

Verity

- ⚙Go to: Start → Run → start „cmd" → "`%windir%\system32\inetsrv\appcmd.exe list config "default web site" /section:requestFiltering`"

Verify the `allowDoubleEscaping` is set to `true`:

- `<requestFiltering allowDoubleEscaping=" true ">`

Config

- ⚙Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ “`notepad web.config`

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

OR

- ⚙ Go to: Start → Run → start „cmd“ → “`%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /allowDoubleEscaping: true`

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

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the Connections pane, select the server, site, application, or directory
- Select Request Filtering
- In the Actions pane, select Edit Feature Settings
- Under the General section, select Allow double escaping

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

HR018: Ensure handler is not granted write and script/execute	✋

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.

Use the CMD shell to configure or verify this setting:

Verity

- ⚙ Go to: Start → Run → start „cmd“ → “`%windir%\system32\inetsrv\ appcmd.exe list config | findstr "handlers"`”

Verify that the `handlers accessPolicy` attribute does not contain `Write` when `Script` or `Execute` are present. The following is an acceptable example:

- `<handlers accessPolicy="Read, Script">`

Config

- ApplicationHost.config (server-wide)
- ⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\inetsrv\appcmd.exe set config /section:handlers /accessPolicy:"[None, Read, Write, Execute, Source, Script, NoRemoteWrite, NoRemoteRead, NoRemoteExecute, NoRemoteScript]"”
- `web.config` (site or application)
- ⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\inetsrv\appcmd.exe set config “default web site” /section:handlers /accessPolicy:"[None, Read, Write, Execute, Source, Script, NoRemoteWrite, NoRemoteRead, NoRemoteExecute, NoRemoteScript]"”

⚠ *This configuration change cannot be made by using IIS Manager.*

HR019: Configure MachineKey validation method	✋

This setting should be deactivated for productive systems.

The `machineKey` element of the ASP.NET `web.config` specifies the algorithm and keys that ASP.NET will use for encryption.

The following encryption methods are available:

- Advanced Encryption Standard (AES)
- Message Digest 5 (MD5)
- Secure Hash Algorithm (SHA1)
- Triple Data Encryption Standard (TripleDES)

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

Use the CMD shell to configure or verify this setting:

⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\inetsrv\appcmd set config /commit:WEBROOT /section:machineKey /validation:SHA1”

Verify the Machine Key encryption method using IIS Manager:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Navigate to the level that was configured, the WEBROOT, or server
- In the features view, select Machine Key
- On the Machine Key page, verify that SHA1 is selected in the Encryption method dropdown

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

HR020: Disable HTTP trace method	✋

Configure this setting only in productive systems.

The HTTP TRACE method returns the contents of client HTTP requests in the entity-body of the TRACE response. Mitigate this by setting the `<verbs>` element of the `<requestFiltering>` collection.

- It is recommended the HTTP TRACE method be denied.

Use the CMD shell to configure or verify this setting:

Verity

- ApplicationHost.config (server-wide)
- ⚙Go to: Start → Run → start „cmd“ → “%windir%\system32\inetsrv\appcmd list config /section:requestFiltering”
- `web.config` (site or application)
- ⚙Go to: Start → Run → start „cmd“ → “%windir%\system32\inetsrv\appcmd list config “default web site" /section:requestFiltering”

Verify that the `add verb` attribute in the `<verbs>` element is set to “"TRACE" allowed="false"”:

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

Config: Add or Del an Allow or Deny Verb

Add

- E.g. `web.config` (site or application)
- ⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /+"verbs.[verb='TRACE',allowed='[false or true]'"

DEL

- E.g. `web.config` (site or application)
- ⚙ Go to: Start → Run → start „cmd“ → “`%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /-"verbs.[verb='TRACE']`”

⚠ To set configurations in `ApplicationHost.config` (server-wide), delete the "`default web site`" string.

Use IIS Manager to set the `add verb` attribute

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Select the site, application, or directory to be configured
- In the Home pane, double-click Request Filtering
- In the Request Filtering pane, click the HTTP verbs tab, and then click Deny Verb... in the Actions pane
- In the Deny Verb dialog box, enter the TRACE

HR021: Allow white-listed file extensions only	✋

The `FileExtensions` Request Filter allows defining specific extensions with the web server 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.

- It is recommended that all extensions be unallowed at the most global level possible, with only those necessary being allowed.

Use the CMD shell to configure or verify this setting:

Verity

- ⚙Go to: Start → Run → start „cmd“ → “`%windir%\system32\inetsrv\appcmd.exe list config "default web site" /section:requestFiltering`”

Verify the `allowDoubleEscaping` is set to `false`:

- `< fileExtensions allowUnlisted="[false or true]">`

Config

The following web.config will disallow any requests for files that do not have .asp, .aspx, or .html as their extension.

⚙Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ “`notepad web.config`

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

```
      </system.webServer>
```

OR

- ⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /fileExtensions.allowUnlisted:[false or true]”

Add or Del an allow or deny File Name Extention

Add

- ⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /+"fileExtensions.[fileExtension='(e.g. exe)',allowed='[false or true]'"

DEL

- ⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\inetsrv\appcmd.exe set config "default web site" /section:requestFiltering /-"fileExtensions.[fileExtension='(e.g. exe)']”

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

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the Connections pane, select the server, site, application, or directory
- Select Request Filtering
- In the Actions pane, select Edit Feature Settings
- Under the General section, uncheck Allow unlisted file name extensions

⚠ *A standard list already exists, which should be completed in extreme cases.*

HR022: Lock down encryption providers	✋

By default, whenever a property is encrypted, IIS uses the “defaultProvider” for encryption defined in “machine.config“. The default value for this is RsaProtectedConfigurationProvider. The IIS local system process (WAS) runs under the context of LOCALSYSTEM and needs access to the application pool passwords. 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.

To verify the permissions have been removed:

1. Obtain the machine GUID.
   ⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\reg.exe query "hklm\software\microsoft\cryptography" | find "MachineGuid"”
2. Next ensuring that the BUILTIN\IIS_IUSRS has been removed.
   ⚙ Go to: Start → Run → start „cmd“ → “%systemroot%\system32\icacls.exe %allusersprofile%\microsoft\crypto\rsa\machinekeys\76944fb33636aeddb9590521c2e8815a_<MachineGUID>”

Removing access to the iisWasKey:

32-bit system

⚙ Go to: Start → Run → start „cmd“ →
```
"%systemroot%\Microsoft.NET\Framework\v2.0.50727\aspnet_regiis.exe -pr iisWasKey
IIS_IUSRS"
```

64-bit system

⚙ Go to: Start → Run → start „cmd“ →
```
"%systemroot%\Microsoft.NET\Framework64\v2.0.50727\aspnet_regiis.exe -pr iisWasKey
IIS_IUSRS"
```

⚠ *The syntax is depending on the version of .NET being used.*

HR023:.Net: Turn Debug Off	✋

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.

- On productive systems debugging must be turned off.

Use the CMD shell to verify this setting:

⚙ Go to: Start → Run → start „cmd“ → navigate to the configured site / application / content location→ `"type web.config | findstr "compilation""`

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

Use the UI to make this change:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- *Navigate desired server, site, or application*
- *In Features View, select .NET Compilation*
- *On the .NET Compilation page, in the Behavior section, ensure the Debug field is set to False*
- In the Actions pane, "Apply" settings

⚠ *This is a defense in depth recommendation due to the* `<deployment retail="true" />` *in the* `machine.config` *configuration file overriding any debug settings.*

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.*

HR024: Configure global authorization rule to restrict access	✋

URL Authorization allows the addition of Authorization Rules to the 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.

- URL Authorization should be configured to only grant access to the necessary security principals.
    - Configure authorization rule on server level to grand access only for *Administrators*

Configure and verify that URL Authorization Rules have been applied using IIS Manager:

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- Select the hierarchy you want to add an Authorization Rule (e.g. on server level select the IIS)
- Select Authorization Rules
- Remove the "Allow All Users" rule
- Click Add Allow or Deny Rule

Verify that URL Authorization Rules have been applied using CMD:

Server level:

⚙ Go to: Start → Run → start „cmd" → `"%systemroot%\system32\inetsrv\appcmd list config /section:system.webserver/security/authorization"`

⚠ *That the above setting sometimes only show up if it was set to something different. To truly verify the setting in the web.config, the setting can be changed, and then changed back.*

HR025: Ensure configuration attribute notListedIsapisAllowed set to false	✋

This element ensures that malicious users cannot copy unauthorized ISAPI binaries to the Web server and then run them.

- `notListedIsapisAllowed` must be set to `false`.

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>`.

Use the CMD shell to configure or verify this setting:

Verity

⚙Go to: Start → Run → start „cmd" → `"%windir%\system32\inetsrv\appcmd list config /section:security/isapiCgiRestriction"`

Verify that the `notListedIsapisAllowed` attribute in the `<isapiCgiRestriction>` element is set to `false`:

- `<isapiCgiRestriction notListedIsapisAllowed="false">`

Config

- `ApplicationHost.config` (server-wide)
- ⚙ Go to: Start → Run → start „cmd“ → “`%systemroot%\system32\inetsrv\ appcmd set config /section:security/isapiCgiRestriction /notListedIsapisAllowed:[false or true]`”

Use IIS Manager to set the `notListedIsapisAllowed` attribute to false:

⚙ Go to: Start → Run → start „control“ →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the Connections pane, select the server
- In Features View, select ISAPI and CGI Restrictions
- In the Actions pane, select Edit Feature Settings
- In the Edit ISAPI and CGI Restrictions Settings dialog, clear the Allow unspecified ISAPI modules check box

HR026: Ensure configuration attribute notListedCgisAllowed set to false	✋

This element ensures that malicious users cannot copy unauthorized CGI binaries to the Web server and then run them.

- `notListedCgisAllowed` must be set to `false`.

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>`.

Use the CMD shell to configure or verify this setting:

Verity

- ⚙Go to: Start → Run → start „cmd“ → “`%windir%\system32\inetsrv\appcmd list config /section:security/isapiCgiRestriction`”

Verify that the `notListedCgisAllowed` attribute in the `<isapiCgiRestriction>` element is set to `false`:

- `<isapiCgiRestriction notListedCgisAllowed ="false">`

Config

- `ApplicationHost.config` (server-wide)
- ⚙ Go to: Start → Run → start „cmd“ → “`%systemroot%\system32\inetsrv\ appcmd set config /section:security/isapiCgiRestriction /notListedCgisAllowed:[false or true]`”

Use IIS Manager to set the `notListedCgisAllowed` attribute to false:

⚙ Go to: Start → Run → start „control" →navigate to Administrative Tools → Internet Information Services (IIS) Manager

- In the Connections pane, select the server
- In Features View, select ISAPI and CGI Restrictions
- In the Actions pane, select Edit Feature Settings
- In the Edit ISAPI and CGI Restrictions Settings dialog, clear the Allow unspecified CGI modules check box

HR027: Encrypt connection string information stored in the web.config	✋

Improve the security of sensitive information stored in a connection string, such as the database name, user name, password, and so on, by encrypting the connection string section of the Web.config file using protected configuration.

- Connection string information should be encrypted.

To encrypt connection string information stored in the Web.config file perform the following:

32-bit system

⚙ Go to: Start → Run → start „cmd" → `"%systemroot%\Microsoft.NET\Framework\v2.0.50727\aspnet_regiis -pe "connectionStrings" -app "/Application""`

64-bit system

⚙ Go to: Start → Run → start „cmd" → `"%systemroot%\Microsoft.NET\Framework64\v2.0.50727\aspnet_regiis -pe "connectionStrings" -app "/Application""`

Verify the contents of the web.config files. The connection strings configuration section should contain encrypted information instead of a clear-text connection string.

⚠ *The syntax is depending on the version of .NET being used.*

4 DOCUMENT HISTORY

Version	Date	Editor(s)	Changes
0.9.4	26.06.2015	Dominik Phillips	- Initial Version
0.9.5	26.06.2015	Niki Vonderwell	- Document Review
1.0.0	26.06.2015	Dominik Phillips	- Initial Version