AlfaSoft Agile Software Company

pentru

Sistemul Informațional Automatizat Cadastrul Fiscal
Versiunea 2

SICF SQL 2017 Post-Installation Procedures

Version D_SICF-SQLPIP-001 ● December 28, 2019

alfasoft
web software company

Last edited: December 28, 2019

Document History

Description	Author	Version	Date
Created initial version	Anatol Bobichev	n/a	December 14 2018
Document formatting reviewed	Victoria Plugari	D_SICF-WSICM-001	December 19, 2018
Document reviewed	Alexandr pascalov	D_SICF-WSICM-001	December 28, 2019

Table of Contents

1 SETTING UP SHARED ASP.NET SESSION STATE 6

1.1 SQL Server Session State Service (Default, "Classic" model) 6

1.2 Local server Session State management should be configured. 6

1.3 SQL Server Session Management Mode (Enterprise Model) 7

1.4 Security Tightening 9

1.4.1 Debug 9

1.4.2 Request Validation 9

1.4.3 Tracing 9

1.4.4 Errors handling 9

1.4.5 Cross site scripting (XSS) vulnerability preventing 9

1.5 Command Injection 11

2 WEB SITE CONFIGURATION FILES ENCRYPTION 12

2.1 Encrypting web.config Files 12

2.2 Decripting cintent of the web.config files 12

2.3 Secure cookies attributes 13

2.4 Add or Change Response Headers in IIS 13

2.5 X-Content-Type-Options 13

2.6 The X-Frame-Options response header 13

3 TROUBLESHOOTING INSTALLATION AND INITIAL SETUP 14

3.1 SQL Server Setup Issues 14

3.2 Description of the SQL Server 2008 log files 15

3.2.1 File Summary.txt 15

3.2.2 File Summary_<%ComputerName%_YYYYMMDD_HHMMSS>.txt 15

3.2.3 File Detail.txt 15

3.2.4 File Detail_ComponentUpdate.txt 16

3.2.5 File Detail_GlobalRules.txt 16

3.2.6 MSI Log Files 16

3.2.7 SystemConfigurationCheck_Report.htm 17

3.3 IIS Related Issues 18

3.3.1 Disable IE "Friendly HTTP error messages" 18

3.3.2 Enable IIS7 detailed errors 19

List of Figures

Figure 1 ASP.NET State Service General Properties *6*
Figure 2 ASP.NET State Service Configuration *7*
Figure 3 Handler Mappings for "Temp" map *11*
Figure 4 IE User Friendly Error *18*
Figure 5 IE Advances Internet Options *19*
Figure 6 Server Error (File or Directory not found) *20*
Figure 7 Server Error in Application "Default Web Site" *21*

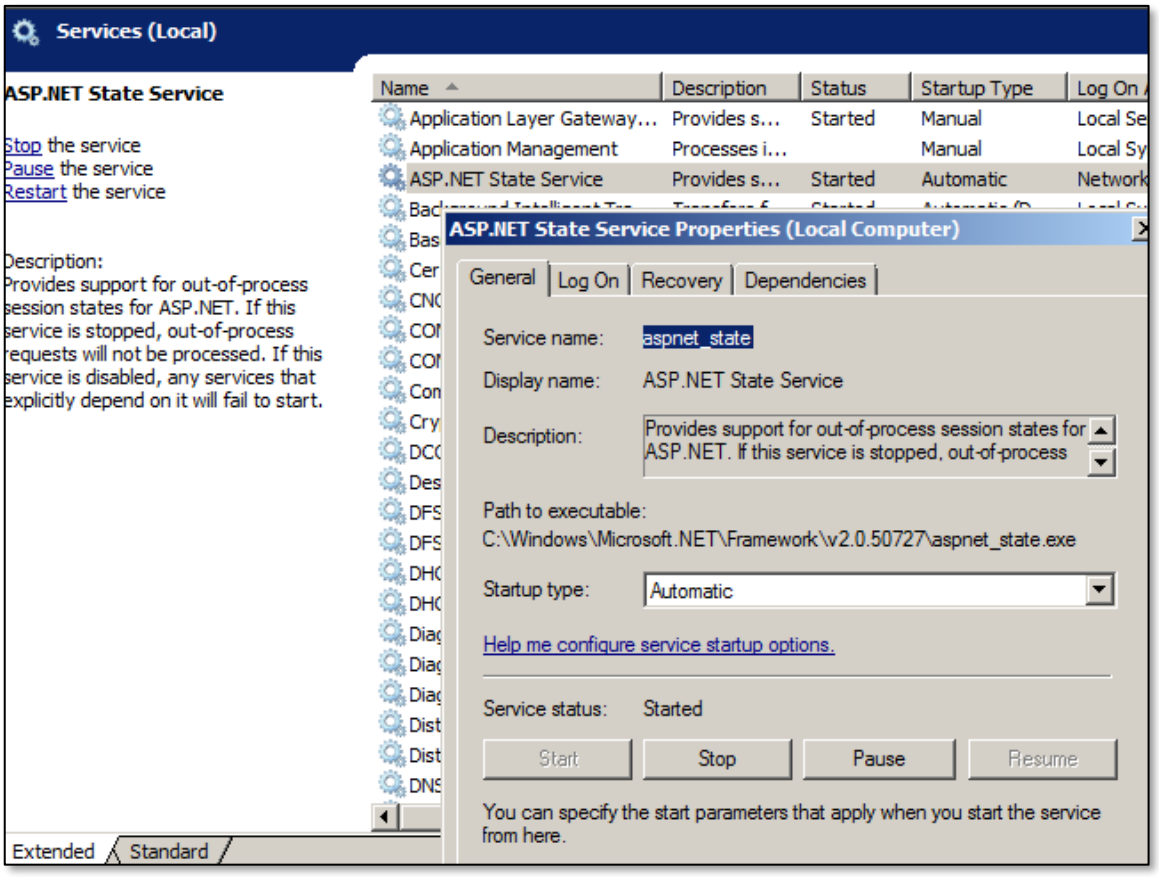
1 Setting Up Shared ASP.NET Session State

1.1 SQL Server Session State Service (Default, “Classic” model)

1.2 Local server Session State management should be configured.

1. Open Start-> Administrative Tools -> Services
2. Locate ASP.NET State Service item in the list (It is stopped and disabled by default).
3. Right-click on ASP.NET State Service item, select "Properties".
4. Configure ASP.NET State Service to "Start Automatically during system startup"

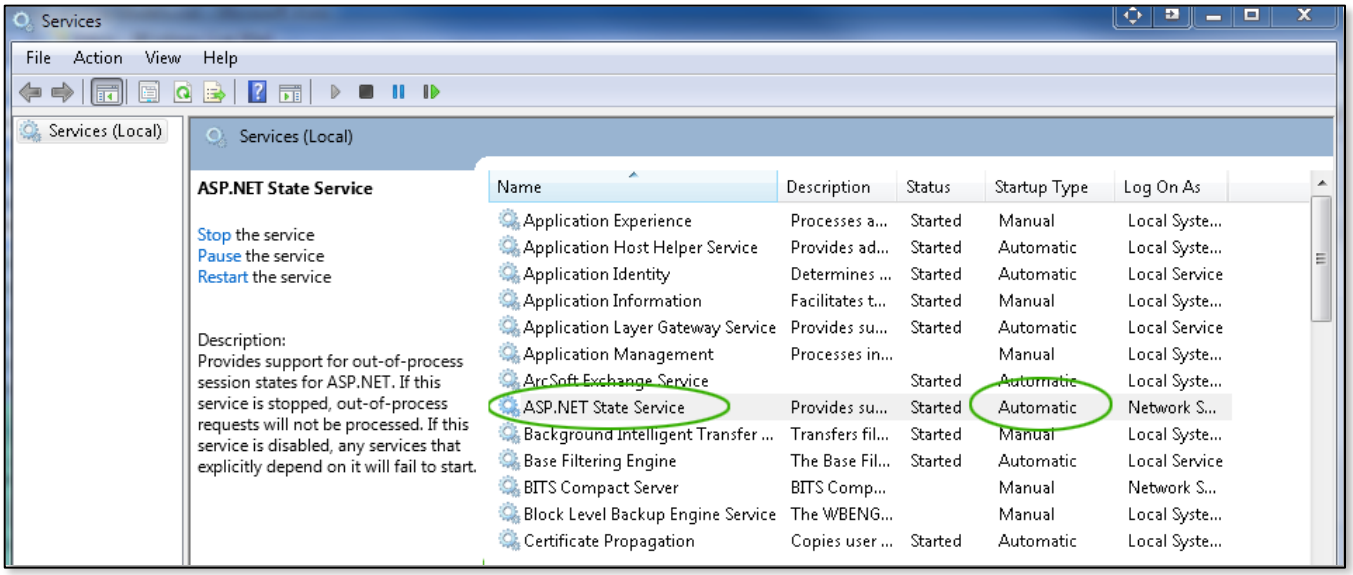
Figure 1 ASP.NET State Service General Properties



ASP.NET State Service Settings.

5. Start ASP.NET State Service if it is stopped, make sure it's running (see screenshot below).

Figure 2 ASP.NET State Service Configuration



1.3 SQL Server Session Management Mode (Enterprise Model)

NOTE: If Application Servers are deployed in clustered server environment, local session state management is not applicable. The Session State for all of the web applications is managed in SQL Server Mode.

SQL Server mode stores session state in a SQL Server database. This ensures that session state is preserved if the Web application is restarted and also makes session state available to multiple Web servers in a Web farm. The SQL Server session state must be configured for each of the applications - Portal, MPass, Notification and Journaling.

1. The command line utility is aspnet_regsql.exe, it must be run with the following parameters:

```
aspnet_regsql -S [server] -E -ssadd -sstype c -d SessionStateDB

// add websitename suffix for each database, eg: SessionStateDBPortal, SessionStateDBMPass, etc.
```

2. The final step is to include the necessary data in to the web.config file. Configure the connection string to the SessionStateDB(for parameter details refer to section **Error! Reference source not found.** above):

```
<sessionState
   mode="SQLServer"
   allowCustomSqlDatabase="true"
```

```
   sqlConnectionString="Data Source= 11.11.11.16;Failover Partner=11.11.11.17;Initial
Catalog=SessionStateDB;UID=UserName;PWD=123;"
   cookieless="false"
   timeout="20"
/>
```

Start Microsoft Internet Explorer and try to access configured site by its domain name If portal code compilation finishes successfully, you should see “Home” page.

1.4 Security Tightening

The following sections do not require any amendments/configuration by administrators and are presented as are in the web site template.

1.4.1 Debug

Compilation should be debug=false on production. No need to change.

```
<compilation debug="false" strict="false" explicit="false" targetFramework="4.0" />
```

1.4.2 Request Validation

Pages **validateRequest="true"** on production. No need to change.

```
<pages validateRequest="true" enableEventValidation="false">
```

1.4.3 Tracing

Trace should be **enabled=false** on production. No need to change.

```
<trace enabled="false" localOnly="false" writeToDiagnosticsTrace="true" />
```

1.4.4 Errors handling

No need to change.

```
<customErrors mode="On" defaultRedirect="~/ServerError.aspx?error=1" />
```

1.4.5 Cross site scripting (XSS) vulnerability preventing

URL filtering prevents servicing the URLs with *script, <script> and* <, > characters. No need to change.

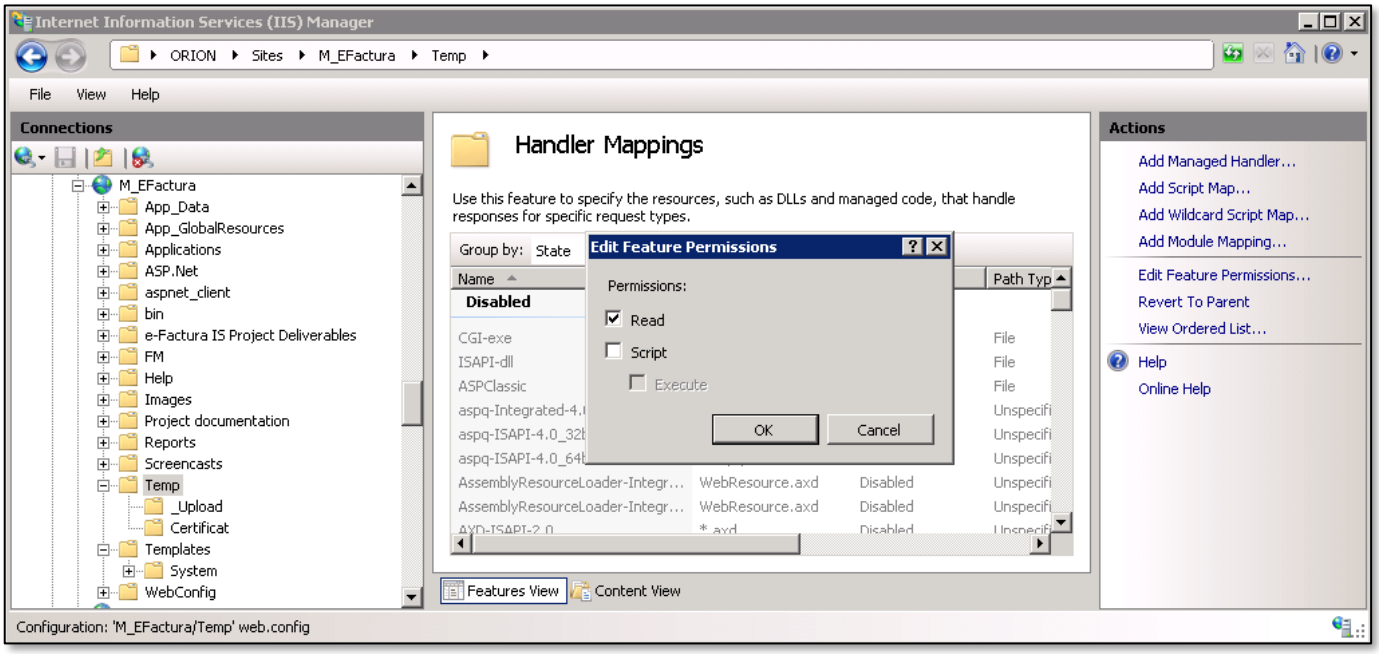
```
<security>
  <requestFiltering allowDoubleEscaping="false" allowHighBitCharacters="true">
        <denyUrlSequences>
        </denyUrlSequences>
        <fileExtensions allowUnlisted="true" />
        <verbs allowUnlisted="true">
        </verbs>
        <denyQueryStringSequences>
```

```
            <add sequence="script" />
            <add sequence="e%3cscript%3e" />
            <add sequence="e%3" />
            <add sequence="%3e" />
            <add sequence="%27" />
    </denyQueryStringSequences>
    <filteringRules>
        </filteringRules>
  </requestFiltering>
</security>
```

1.5 Command Injection

Uploaded files represent a significant risk to applications. The first step in many attacks is to inject some code to the system, which will be attacked. Then the attacker only needs to find a way to get the code executed. Using a file upload helps the attacker accomplish the first step. To secure TEMP storage you must to set Handler Mappings for the map “Temp” that is in the root of the site.

Figure 3 Handler Mappings for “Temp” map



2 Web Site Configuration Files Encryption

2.1 Encrypting web.config Files

Run the following commands in the cmd shell amending the Sitename folder and version of the Framework currently in use (is bolded) in order to encrypt the sensitive information in web.config.

```
CD C:\Windows\Microsoft.NET\Framework\v4.0.30319


aspnet_regiis -pef connectionStrings C:\inetpub\SITENAME -prov
"DataProtectionConfigurationProvider"


aspnet_regiis -pef appSettings C:\inetpub\SITENAME -prov
"DataProtectionConfigurationProvider"


aspnet_regiis -pef system.web/sessionState C:\inetpub\SITENAME -prov
"DataProtectionConfigurationProvider"
```

2.2 Decripting cintent of the web.config files

The following commands will decrypt the web.config file and require similar parameters as encrypting.

```
CD C:\Windows\Microsoft.NET\Framework\v4.0.30319


aspnet_regiis -pdf connectionStrings C:\inetpub\SITENAME
aspnet_regiis -pdf appSettings C:\inetpub\SITENAME
aspnet_regiis -pdf system.web/sessionState C:\inetpub\SITENAME
```

NOTE: Note: As web.config file encryption use internal system certificates, encrypted files can be decrypted back only on server, where they were encrypted initially.

2.3 Secure cookies attributes

Cookies are often a key attack vector for malicious users (typically targeting other users) and, as such, the application should always take due diligence to protect cookies. Certain attributes can be configured, that will add reasonable protection to cookies in front of certain attacks, or will add an additional layer of security to their protection.

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

2.4 Add or Change Response Headers in IIS

2.5 X-Content-Type-Options

The script and styleSheet elements will reject responses with incorrect MIME types if the server sends the response header "X-Content-Type-Options: nosniff". This is a security feature that helps prevent attacks based on MIME-type confusion.

This change impacts the browser's behavior when the server sends the "X-Content-Type-Options: nosniff" header on its responses.

SET X-Content-Type-Options: **NOSNIFF**

Result:

Ensure that in any response received with the "nosniff" directive has a MIME type that matches one of the values listed previously.

2.6 The X-Frame-Options response header

The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page in a <frame> or <iframe>.

SET X-Frame-Options: **DENY**

Result:

Sites can use this to avoid click-*jacking attacks, by ensuring that their content is not embedded into other sites

3 Troubleshooting Installation and initial Setup

3.1 SQL Server Setup Issues

When you run SQL Server Setup, log files are created in a new timestamped log folder. The log folder is located at%programfiles%\Microsoft SQL Server\100\Setup Bootstrap\Log\. The time-stamped log folder name is in the formatYYYYMMDD_hhmmss.

To troubleshoot issues the following files must be reviewed:

- ConfigurationFile.ini
- Summary.txt
- Detail.txt
- Detail_ComponentUpdate.txt
- Sql_common_core_Cpu<32 and or 64>_1.log
- Sql_common_core_loc_Cpu<32 and or 64>_1033_1.log
- Sql_engine_core_inst_Cpu<32 and or 64>_1.log
- Sql_engine_core_inst_loc_Cpu<32 and or 64>_1033_1.log
- Sql_engine_core_shared_Cpu<32 and or 64>_1.log
- Sql_engine_core_shared_loc_Cpu<32 and or 64>_1033_1.log
- Sql_tools_Cpu<32 and or 64>_1.log
- Sql_tools_loc_Cpu<32 and or 64>_1033_1.log
- SqlBrowser_Cpu32_1.log
- Sqlncli_Cpu<32 and or 64>_1.log

Note: In your environment, the file name and number of the log files might differ from the list based on the installation features that are selected when you install SQL Server 2008.

3.2 Description of the SQL Server 2008 log files

3.2.1 File Summary.txt

Location: %ProgramFiles%\Microsoft SQL Server\100\Setup Bootstrap\Log\

Purpose: This log is the clean and user friendly log file that contains the basic information about the problem. This file shows which SQL server components were detected, the operating system environment, the command-line parameters that were specified, the values that were specified, and the overall status of each MSI and MSP file that was executed.

Troubleshooting: To find errors in this file, search for "error" or "failed" keywords. For more information about details of the failure, open the file that is listed in the line starting with "Log with failure".

3.2.2 File Summary_<%ComputerName%_YYYYMMDD_HHMMSS>.txt

Location: %ProgramFiles%\Microsoft SQL Server\100\Setup Bootstrap\Log\<YYYYMMDD_HHMMSS>\Summary_<%ComputerName%_YYYYMMDD_HHMM>txt

Purpose: This log file contains the same information as the Summary.txt file. This log might also contain information about any previous SQL Server Setup attempt.

Troubleshooting: To find errors in this file, you can generally search for "error" or "failed" keywords. For more information about details of the failure, open the file that is listed in the line starting with "Log with failure".

3.2.3 File Detail.txt

Location: %ProgramFiles%\Microsoft SQL Server\100\Setup Bootstrap\Log\<YYYYMMDD_HHMM>\Detail.txt

Purpose: This log file provides a detailed log of the execution. It is organized on lines that begin with the time stamp followed by the extension that produced the log. This is one of the most important log files because it can be used to identify the failures that occur. The logs are generated on a time basis. This means that the actions are logged here not by the component that generated them but by the time at which they are invoked. This is useful to determine the execution process step by step, the order in which actions are executed, and the dependencies between actions. This file is generated for the main workflow such as the install, or the upgrade workflow.

Troubleshooting: If an error occurs in the setup process, the exception or error will be logged at the end of the file. To locate errors in this file, you must go to the end of the file and read the actions in reverse from bottom to top to find where the exception or error is logged. Find the line that starts with "Exit facility code" in the Summary.txt file, and then search for "error," "Watson bucket," or "exception" keywords in the Detail.txt file. The search results near the exit facility code will help you identify when and where the error first occurred.

3.2.4 File Detail_ComponentUpdate.txt

Location: *%ProgramFiles%*\Microsoft SQL Server\100\Setup Bootstrap\Log\<*YYYYMMDD_HHMMSS*>\Detail_ComponentUpdate.txt

Purpose: This log file resembles the Detail.txt file. This file is generated for the component update workflow.

Troubleshooting: Follow the same steps as those in the Detail.txt section.

3.2.5 File Detail_GlobalRules.txt

Location: *%ProgramFiles%*\Microsoft SQL Server\100\Setup Bootstrap\Log\<*YYYYMMDD_HHMMSS*>\Detail_GlobalRules.txt

Purpose: This log file resembles the Detail.txt file. This file is generated for the global rules execution.

Troubleshooting: This log file is used only if the Summary.txt file shows failures in the SystemConfigurationCheck section. Generally, you can open and use the SystemConfigurationCheck_Report.htm file to obtain all the information that is needed. However, if you want additional information from this log, search for the "Rule evaluation done : Failed" phrase in this file.

3.2.6 MSI Log Files

<Feature>_<Architecture>_<Iteration>.log

<Feature>_<Architecture>_<Language>_<Iteration>.log

<Feature>_<Architecture>_<Iteration>_<workflow>.log

Location: *%ProgramFiles%*\Microsoft SQL Server\100\Setup Bootstrap\Log\<*YYYYMMDD_HHMMSS*>\<*Name*>.log

Purpose: These log files provide a detailed log of the package installation process. These logs are generated by the Msiexec.exe process when you install the specified package. If the setup was successful and no errors were encountered, the log entry "Product: *Product_Name* - Installation completed successfully" displays.

Troubleshooting: When you use the MSI log files, make sure that you are looking at the original source of the problem instead of the reactive error messages. To do this, follow these steps:

1. Sort the MSI logs in the same directory by the changed date.
2. Open each MSI log file from the bottom of the log files to each previous log. For each log file, search for "Return value 3" or "@Microsoft" without the quotation marks.
3. Note the error message for each file until you find no error at the end of the logs. The last log contains the original error. When you find the first occurrence in the log, you must determine whether the entries are valid because not all "Return value 3" errors are problems. Some of those errors are expected. If you can't determine whether the errors are valid, we recommend that you try one of the Microsoft SQL Server support options.

3.2.7 SystemConfigurationCheck_Report.htm

Location: *%ProgramFiles%*\Microsoft SQL Server\100\Setup Bootstrap\Log\<*YYYYMMDD_HHMMSS*>\SystemConfigurationCheck_Report.htm

Purpose: This file contains a friendly version of the rules execution status. It also provides a short description of each executed rule.

Troubleshooting: You can open the SystemConfigurationCheck_Report.htm file and look for the "Failed" keyword and examine if any "Warning" entries apply to your environment.

3.3 IIS Related Issues

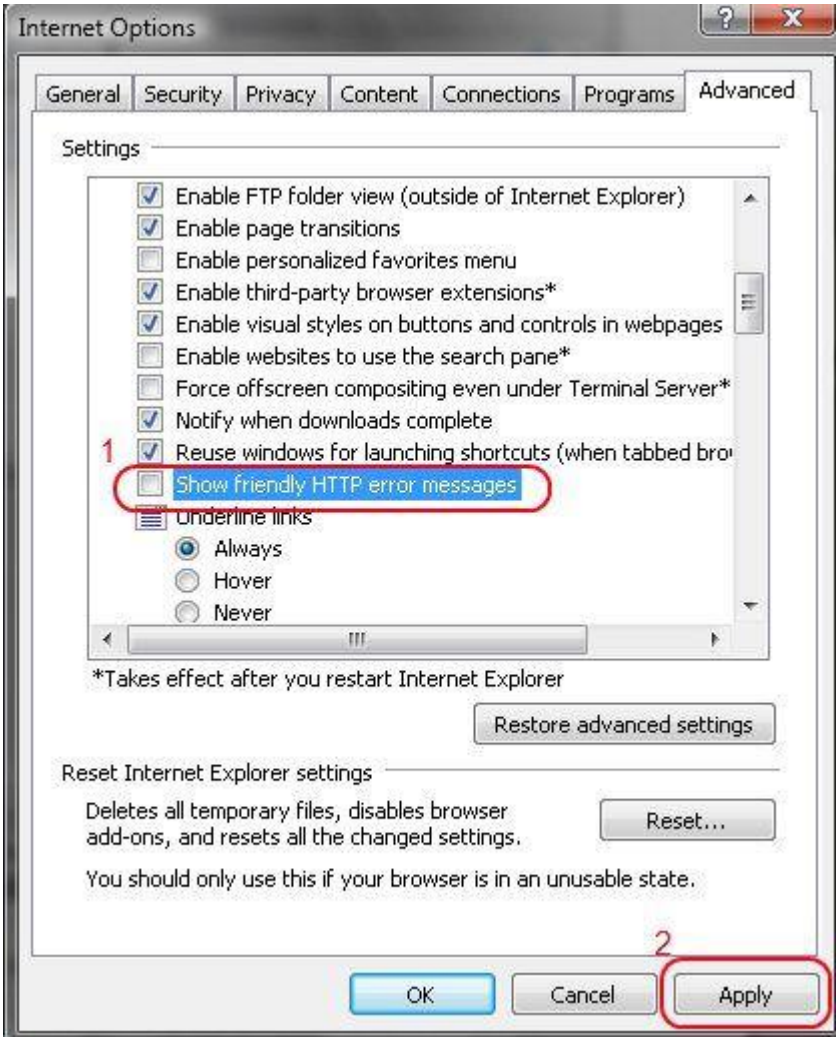
3.3.1 Disable IE "Friendly HTTP error messages"

IE will by default replace the actual error messages coming from the server with a "friendly" error message, which hides the error contents we need to see. For example, for a 404 Not Found error, you may instead see the following:

Figure 4 IE User Friendly Error

To disable this and see the real error coming from the server, you need to go to "Tools > Internet Options", choose the Advanced tab, and clear the "Show friendly HTTP error messages" checkbox. Then, close the browser, open it again, and re-request the page.

Figure 5 IE Advances Internet Options



3.3.2 Enable IIS7 detailed errors

WARNING: This troubleshooting step enables data disclosure vulnerability! Make sure you disable the detailed errors, when obtained the error text.

IIS7 introduces a new custom errors feature, which by default hides the error responses issued by the server to remote clients, replacing them with a basic error message. This is critical for security of your site, as errors frequently contain sensitive information that you don't want others to see, but makes getting to the bottom of the problem harder since you cannot see those very error details. So, if you are requesting your site from another machine, you may still get a basic error that looks like this:

Figure 6 Server Error (File or Directory not found)

There are two options here:

1. **Make the request locally from the server machine.**

By default, you will get the detailed error.

2. **Enable detailed errors for remote clients.**

First, if your error is an ASP.NET exception (you can tell if it says "Runtime Error" and has the framework version), please be aware that ASP.NET overrides the IIS custom error mechanism with its own implementation of custom errors, so you should turn the ASP.NET custom errors off to see detailed ASP.NET exceptions. You DO NOT have to configure IIS7 custom errors for ASP.NET exceptions (it would be silly to have to do it in two places). To turn off ASP.NET custom errors, place the following in your web.config:

```
<system.web>
  <customErrors                    mode="Off"                    />
</system.web>
```

If the error is not an ASP.NET error, turning off IIS7 custom errors will allow error responses from your application to be sent to remote clients without being censored by the IIS7's custom errors module.

Now, you should be getting detailed errors back:

Figure 7 Server Error in Application "Default Web Site"