# 1 Basic Configurations

This section contains basic Web server-level recommendations.

|  |  |  |  |
| --- | --- | --- | --- |
| Control Reference ID | Control Name & Description | Description | Remediation |
| 1.1 | (L1) Ensure 'Web content' is on non-system partition (Manual) | 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. | 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 |
| 1.2 | (L1) Ensure 'Host headers' are on all sites (Automated) | 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. Note: Wildcard host headers are now supported. | Obtain a listing of all sites by using the following appcmd.exe command: Enter the following command in AppCmd.exe to configure the host header: %systemroot%\system32\inetsrv\appcmd.exe set config - section:system.applicationHost/sites /"[name='<website name>'].bindings.[protocol='http',bindingInformation='\*:80:<host header>'].bindingInformation:"\*:80:<host header>"" /commit:apphost OR Enter the following command in PowerShell to configure the host header: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter 'system.applicationHost/sites/site[@name='<website name>']/bindings/binding[@protocol='http' and @bindingInformation='\*:80:']' - name 'bindingInformation' -value '\*:80:<host header value>' OR 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. |
| 1.3 | (L1) Ensure 'Directory browsing' is set to Disabled (Automated) | 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 Note: If directory browsing is enabled (an exception to this recommendation), make sure that it is only enabled on the particular directory or directories that need to be shared. | 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: Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config /section:directoryBrowse /enabled:false OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -Filter system.webserver/directorybrowse -PSPath iis:\ -Name Enabled -Value False |
| 1.4 | (L1) Ensure 'application pool identity' is configured for all application pools (Automated) | 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. IIS 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. To achieve isolation in IIS, 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. 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 unless there is an underlying reason that the application pool needs to run as a specified end user account. One example where this is needed is for web farms using Kerberos authentication. | 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 ApplicationPool identity to the built-in ApplicationPoolIdentity using AppCmd.exe, run the following from a command prompt: Enter the following command in AppCmd.exe to configure %systemroot%\system32\inetsrv\appcmd set config /section:applicationPools /[name='<apppool name>'].processModel.identityType:ApplicationPoolIdentity OR To change the ApplicationPool identity to the built-in ApplicationPoolIdentity using PowerShell: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter 'system.applicationHost/applicationPools/add[@name='<apppool name>']/processModel' -name 'identityType' -value '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. 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. |
| 1.5 | (L1) Ensure 'unique application pools' is set for sites (Automated) | Application Pool Identities 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. | The following appcmd.exe command will set the application pool for a given application: %systemroot%\system32\inetsrv\appcmd set app '<website name>/' /applicationpool:<apppool name> The output of this command will be similar to the following: APP object "Default Web Site/" changed (applicationPool:DefaultAppPool) Run the above command to ensure a unique application pool is assigned for each site listed OR Enter the following command in PowerShell to configure: Set-ItemProperty -Path 'IIS:\Sites\<website name>' -Name applicationPool - Value <apppool name> OR 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 |
| 1.6 | (L1) Ensure 'application pool identity' is configured for anonymous user identity (Automated) | To achieve isolation in IIS, 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. | 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 OR To use AppCmd.exe to configure anonymousAuthentication at the server level, the command would look like this: %systemroot%\system32\inetsrv\appcmd set config - section:anonymousAuthentication /username:"" --password OR Enter the following command in PowerShell to configure: Set-ItemProperty -Path IIS:\AppPools\<apppool name> -Name passAnonymousToken -Value True |
| 1.7 | (L1) Ensure' WebDav' feature is disabled (Automated) | WebDAV is an extension to the HTTP protocol which allows clients to create, move, and delete files and resources on the web server. Note: The WebDAV feature must be enabled for this functionality to be available in IIS. | To disable this feature using PowerShell, enter the following command: Uninstall-WindowsFeature Web-DAV-Publishing Verify that Success is True |

# 2 Configure Authentication and Authorization

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

|  |  |  |  |
| --- | --- | --- | --- |
| Control Reference ID | Control Name & Description | Description | Remediation |
| 2.1 | (L1) Ensure 'global authorization rule' is set to restrict access (Manual) | IIS 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. | To configure URL Authorization at the server level using command line utilities: Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config - section:system.webServer/security/authorization /- "[users='\*',roles='',verbs='']" %systemroot%\system32\inetsrv\appcmd set config - section:system.webServer/security/authorization /+"[accessType='Allow',roles='Administrators']" OR Enter the following command in PowerShell to configure: Remove-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/authorization" -name "." -AtElement @{users='\*';roles='';verbs=''} Add-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/authorization" -name "." -value @{accessType='Allow';roles='Administrators'} OR 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 Administrators group) |
| 2.2 | (L1) Ensure access to sensitive site features is restricted to authenticated principals only (Manual) | IIS 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 redirectionbased 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 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 username and password which is then validated against a database or other credentials store • Basic authentication - requires a valid username 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 nonpublic web services be configured with a credentials-based authentication mechanism. | When configuring an authentication module for the first time, each mechanism must be completely configured before use. 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 OR Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config - section:system.web/authentication /mode:<Windows|Passport|Forms> OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -location '<website location>' -filter 'system.webServer/security/authentication/anonymousAuthentication' -name 'enabled' -value 'False' Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -location '<website location>' -filter 'system.webServer/security/authentication/windowsAuthentication' -name 'enabled' -value 'True' |
| 2.3 | (L1) Ensure 'forms authentication' require SSL (Automated) | 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. NOTE Due to identified security vulnerabilities, SSL no longer provides adequate protection for a sensitive information. | 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 OR Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config - section:system.web/authentication /mode:Forms OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/Default Web Site' -filter 'system.web/authentication/forms' -name 'requireSSL' -value 'True' |
| 2.4 | (L2) Ensure 'forms authentication' is set to use cookies (Automated) | 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. | 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 OR Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config - section:system.web/authentication /forms.cookieless:"UseCookies" OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/Default Web Site' -filter 'system.web/authentication/forms' -name 'cookieless' -value 'UseCookies' |
| 2.5 | (L1) Ensure 'cookie protection mode' is configured for forms authentication (Automated) | 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. | 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 OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/<website name>' -filter 'system.web/authentication/forms' -name 'protection' -value 'All' |
| 2.6 | (L1) Ensure transport layer security for 'basic authentication' is configured (Automated) | Basic Authentication can pass credentials across the network in clear text. It is therefore imperative that the traffic between client and server be encrypted, especially in cases where the site is publicly accessible and is recommended that TLS be configured and required for any Site or Application using Basic Authentication. | To protect Basic Authentication with transport layer security: 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 X.509 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. 3. In the Actions pane, click Apply OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -location '<website name>' -filter 'system.webServer/security/access' -name 'sslFlags' -value 'Ssl' |
| 2.7 | (L1) Ensure 'passwordFormat' is not set to clear (Automated) | The <credentials> element of the <authentication> element allows optional definitions of name and password for IIS Manager User accounts within the configuration file. Forms based authentication also uses these elements to define the users. IIS Manager Users can use the administration interface to connect to sites and applications in which they've been granted authorization. Note: 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. | 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 The clear text passwords will need to be replaced with the appropriate hashed version. OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/<website name>' -filter 'system.web/authentication/forms/credentials' -name 'passwordFormat' -value 'SHA1' |
| 2.8 | (L2) Ensure 'credentials' are not stored in configuration files (Automated) | The <credentials> element of the <authentication> element allows optional definitions of name and password for IIS Manager User accounts within the configuration file. Forms based authentication also uses these elements to define the users. IIS Manager Users can use the administration interface to connect to sites and applications in which they've been granted authorization. Note: The <credentials> element only applies when the default provider, ConfigurationAuthenticationProvider, is configured as the authentication provider. It is recommended to avoid storing passwords in the configuration file even in form of hash. | 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, remove the section This will remove all references to stored users in the configuration files. OR Enter the following command in PowerShell to configure: Remove-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/<website name>' -filter 'system.web/authentication/forms/credentials' -name '.' |

# 3 ASP.NET Configuration Recommendations

This section contains recommendations specific to ASP.NET.

|  |  |  |  |
| --- | --- | --- | --- |
| Control Reference ID | Control Name & Description | Description | Remediation |
| 3.1 | (L1) Ensure 'deployment method retail' is set (Manual) | 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. | 1. Open the machine.config file located in: %systemroot%\Microsoft.NET\Framework<bitness (if not the 32 bit)>\<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: %systemroot%\Microsoft.NET\Framework<bitness (if not the 32 bit)>\<framework version>\CONFIG |
| 3.2 | (L2) Ensure 'debug' is turned off (Automated) | 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. | 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. OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/<website name>' -filter "system.web/compilation" -name "debug" -value "False" |
| 3.4 | (L1) Ensure IIS HTTP detailed errors are hidden from displaying remotely (Automated) | 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. | 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 OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/<website name>' -filter "system.webServer/httpErrors" -name "errorMode" -value "DetailedLocalOnly" |
| 3.5 | (L2) Ensure ASP.NET stack tracing is not enabled (Automated) | 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. | 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: <configuration> <system.web> <trace enabled="true"> </system.web> </configuration> OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/<website name>' -filter "system.web/trace" -name "enabled" -value "False" |
| 3.6 | (L2) Ensure 'httpcookie' mode is configured for session state (Automated) | 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. | 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: %systemroot%\system32\inetsrv\appcmd set config /commit:WEBROOT /section:sessionState /cookieless:UseCookies /cookieName:ASP.NET\_SessionID /timeout:20 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. OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/<website name>' -filter "system.web/sessionState" -name "mode" -value "StateServer" |
| 3.7 | (L1) Ensure 'cookies' are set with HttpOnly attribute (Automated) | 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. | 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> 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. All modern versions of browsers recognize HttpOnly attribute; older versions will either treat them as normal cookies or simply ignore them altogether. |
| 3.8 | (L2) Ensure 'MachineKey validation method - .Net 3.5' is configured (Automated) | 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 validation 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. | 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, the /commit:WEBROOT switch must be included so that configuration changes are made to the root web.config file instead of ApplicationHost.config. |
| 3.9 | (L1) Ensure 'MachineKey validation method - .Net 4.5' is configured (Automated) | 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 validation 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 • Secure Hash Algorithm (SHA-2) is a family of two similar hash functions, with different block sizes known as SHA-256 and SHA-512. They differ in the word size; SHAS-256 used 32-bit words and SHA-512 uses 64-bit words. It is recommended that SHA-2 methods be configured for use at the global level. | 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:<validation method> Note: When Appcmd.exe is used to configure the <machineKey> 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. OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT' -filter "system.web/machineKey" -name "validation" -value "<validation method>" |
| 3.12 | (L2) Ensure Server Header is removed (Manual) | The server header headers specify the underlying technology used by the application. | Enter the following command to use AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd.exe set config - section:system.webServer/security/requestFiltering /removeServerHeader:"True" /commit:apphost OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST/' -filter "system.webServer/security/requestFiltering" -name "removeServerHeader" - value "True" |

## 3.3 Configure Data Access Control Lists

Configure data access control lists based on a user’s need to know. Apply data access control lists, also known as access permissions, to local and remote file systems, databases, and applications.  
● ● ●

## 3.10 Encrypt Sensitive Data in Transit

Encrypt sensitive data in transit. Example implementations can include: Transport Layer Security (TLS) and Open Secure Shell (OpenSSH).

## 3.11 Encrypt Sensitive Data at Rest

Encrypt sensitive data at rest on servers, applications, and databases containing sensitive data. Storage-layer encryption, also known as server-side encryption, meets the minimum requirement of this Safeguard. Additional encryption methods may include application-layer encryption, also known as client-side encryption, where access to the data storage device(s) does not permit access to the plain-text data.  
● ●  
v7

# 4 Request Filtering and Other Restriction Modules

Introduced in IIS 7.0 for the first time, Request Filtering is a powerful module that 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. 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.  
IIS 8 also introduced modules for Dynamic IP Address Restrictions. This module can be configured to automatically block web site access based on specific rules.  
Note: Request Filtering and IP and Domain Restrictions must be enabled as a role service under IIS in order to configure any of its features.

|  |  |  |  |
| --- | --- | --- | --- |
| Control Reference ID | Control Name & Description | Description | Remediation |
| 4.1 | (L2) Ensure 'maxAllowedContentLength' is configured (Manual) | 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. | 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) Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config /section:requestfiltering /requestLimits.maxAllowedContentLength:30000000 OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/requestFiltering/requestLimits" -name "maxAllowedContentLength" -value 30000000 |
| 4.2 | (L2) Ensure 'maxURL request filter' is configured (Automated) | 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 URL. | 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 Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config /section:requestfiltering /requestLimits.maxURL:4096 OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/requestFiltering/requestLimits" -name "maxUrl" - value 4096 |
| 4.3 | (L2) Ensure 'MaxQueryString request filter' is configured (Automated) | 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 that can be accepted in the query string. | 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 Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config /section:requestfiltering /requestLimits.maxQueryString:2048 OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/requestFiltering/requestLimits" -name "maxQueryString" -value 2048 |
| 4.4 | (L2) Ensure non-ASCII characters in URLs are not allowed (Automated) | This feature is used to allow or reject all requests to IIS 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. | 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. Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config /section:requestfiltering /allowHighBitCharacters:false OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/requestFiltering" -name "allowHighBitCharacters" - value "False" |
| 4.5 | (L1) Ensure Double-Encoded requests will be rejected (Automated) | 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 doubleencoded 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. The double-encoded requests filter was the VerifyNormalization option in UrlScan. It is recommended that double-encoded requests be rejected. | 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. Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config /section:requestfiltering /allowDoubleEscaping:false OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/requestFiltering" -name "allowDoubleEscaping" - value "True" |
| 4.6 | (L1) Ensure 'HTTP Trace Method' is disabled (Manual) | The HTTP TRACE method returns the contents of client HTTP requests in the entitybody 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. The <verbs> element replaces the [AllowVerbs] and [DenyVerbs] features in UrlScan. It is recommended the HTTP TRACE method be denied. | 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 Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config /section:requestfiltering /+verbs.[verb='TRACE',allowed='false'] OR Enter the following command in PowerShell to configure: Add-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/requestFiltering/verbs" -name "." -value @{verb='TRACE';allowed='False'} |
| 4.7 | (L1) Ensure Unlisted File Extensions are not allowed (Automated) | 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 disallowed at the most global level possible, with only those necessary being allowed. | 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 Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config /section:requestfiltering /fileExtensions.allowunlisted:false OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/requestFiltering/fileExtensions" -name "allowUnlisted" -value "False" |
| 4.9 | (L1) Ensure 'notListedIsapisAllowed' is set to false (Automated) | 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. | 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 Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd.exe set config - section:system.webServer/security/isapiCgiRestriction /notListedIsapisAllowed:false OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/isapiCgiRestriction" -name "notListedIsapisAllowed" -value "False" |
| 4.10 | (L1) Ensure 'notListedCgisAllowed' is set to false (Automated) | 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. | 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 Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd.exe set config - section:system.webServer/security/isapiCgiRestriction /notListedCgisAllowed:false OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/isapiCgiRestriction" -name "notListedCgisAllowed" -value "False" |
| 4.11 | (L1) Ensure 'Dynamic IP Address Restrictions' is enabled (Manual) | Dynamic IP address filtering allows administrators to configure the server to block access for IPs that exceed the specified number of requests or request frequency. Note: Ensure that you receive the Forbidden page once the block has been enforced. | 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. OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/dynamicIpSecurity/denyByConcurrentRequests" -name "enabled" -value "True" Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.webServer/security/dynamicIpSecurity/denyByConcurrentRequests" -name "maxConcurrentRequests" -value <number of requests> |

## 4.8 Uninstall or Disable Unnecessary Services on

Enterprise Assets and Software Uninstall or disable unnecessary services on enterprise assets and software, such as an unused file sharing service, web application module, or service function.  
● ●

# 5 IIS Logging Recommendations

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

|  |  |  |  |
| --- | --- | --- | --- |
| Control Reference ID | Control Name & Description | Description | Remediation |
| 5.1 | (L1) Ensure Default IIS web log location is moved (Automated) | 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 recommended that the default location for IIS log files be changed to a restricted, non-system drive. | Moving the default log location can be easily accomplished using the Logging feature in the IIS Management UI, AppCmd.exe, or PowerShell. Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd set config -section:sites - siteDefaults.logfile.directory:<new log location> OR Enter the following command in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.applicationHost/sites/siteDefaults/logFile" -name "directory" -value <new log location> 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. 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. |
| 5.2 | (L1) Ensure Advanced IIS logging is enabled (Automated) | 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 clientside fields can be easily logged with minor configuration in the IIS management console. | 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 Logging icon on the Home page 4. Click Select Fields The fields that will be logged need to be configured using the Add or Edit Fields button. Note: There may be performance considerations depending on the extent of the configuration. |
| 5.3 | (L1) Ensure 'ETW Logging' is enabled (Manual) | Event Tracing for Windows (ETW) is a Windows feature that allows Administrators to send logging information to another location. This information is then compiled on the server and can be queried. | 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. |

# 6 FTP Requests

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

|  |  |  |  |
| --- | --- | --- | --- |
| Control Reference ID | Control Name & Description | Description | Remediation |
| 6.1 | (L1) Ensure FTP requests are encrypted (Manual) | FTP Publishing Service for IIS 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. | To configure FTP over SSL at the server level using AppCmd.exe or PowerShell: Enter the following command in AppCmd.exe to configure: %systemroot%\system32\inetsrv\appcmd.exe set config - section:system.applicationHost/sites /siteDefaults.ftpServer.security.ssl.controlChannelPolicy:"SslRequire" /siteDefaults.ftpServer.security.ssl.dataChannelPolicy:"SslRequire" /commit:apphost OR Enter the following commands in PowerShell to configure: Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.applicationHost/sites/siteDefaults/ftpServer/security/ssl" -name "controlChannelPolicy" -value "SslRequire" Set-WebConfigurationProperty -pspath 'MACHINE/WEBROOT/APPHOST' -filter "system.applicationHost/sites/siteDefaults/ftpServer/security/ssl" -name "dataChannelPolicy" -value "SslRequire" |

## 6.2 Activate audit logging

Ensure that local logging has been enabled on all systems and networking devices.  
● ● ●

## 6.3 Enable Detailed Logging

Enable system logging to include detailed information such as an event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.  
● ●

## 6.4 Ensure adequate storage for logs

Ensure that all systems that store logs have adequate storage space for the logs generated.  
● ●

## 6.6 Deploy SIEM or Log Analytic tool

Deploy Security Information and Event Management (SIEM) or log analytic tool for log correlation and analysis.

## 6.8 Define and Maintain Role-Based Access Control

Define and maintain role-based access control, through determining and documenting the access rights necessary for each role within the enterprise to successfully carry out its assigned duties. Perform access control reviews of enterprise assets to validate that all privileges are authorized, on a recurring schedule at a minimum annually, or more frequently.  
●  
v7

# 7 Transport Encryption

This section contains recommendations for configuring IIS protocols and cipher suites.  
For security protocols (SSL, TLS), there are 2 registry paths that control a protocol state in the O/S: TLS client and TLS server. A web server normally acts as the TLS server in that it is serving web content to clients. There are some instances where a web server is configured as a 'client'. An example of a server acting as a client can be seen when there is dynamic content generation. The webserver queries a remote database server to return content specific to a user's request. In this configuration, the web server is acting as a TLS client. In cases such as these, the configured TLS server protocol and cipher suite preferences take precedence over the client's. This behavior is why for the IIS benchmark we require specific protocol settings for a TLS server and only recommend settings for TLS clients.  
If SSLv3 registry keys are not set, the O/S defaults take precedence.  
For example, to disable SSLv3 protocol on the TLS server, you need to set the following registry key to 0: HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 3.0\Server\Enabled  
To prevent a client from issuing the Hello command over that legacy protocol the following registry must be set to 0: HKLM\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 3.0\Client\Enabled  
The fact that the key is named Enabled can be confusing. The setting of the value to either 0 or 1 actually sets the state of the protocol. 0 being disabled and 1 being enabled.  
Here are some specifics into how "Enabled" and "DisabledByDefault" registry settings work. The following article, How to restrict the use of certain cryptographic algorithms and protocols in Schannel.dll, provides additional information related to controlling these protocols and ciphers.  
Using the "Enabled = 0" registry setting disables the protocol in a way that can't be overridden by application settings. This is the only robust way to prevent the protocol from being used and no additional settings are required. At the same time, using the "DisabledByDefault" registry setting only prevents that protocol from issuing the Hello command over that protocol when an SSL connection with a server is initiated. This O/S level setting can be overridden by an application which has application specific TLS coding. An example of this can be shown by setting the protocol within a line of code in your .Net 4.5 application: ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12. This can override the O/S setting if the DisabledByDefault key is present. "DisabledByDefault" is useful in the case when you want to have some control over the system settings but also allow an application to explicitly specify the protocols they would like to use.  
Enabled only works strongly in the negative case ("Enabled = 0"). If "Enabled=1" or is not set, then "DisabledByDefault" will override in the case where the application takes the system defaults. "Enabled=1" is also overridden by application specific protocol flags.

|  |  |  |  |
| --- | --- | --- | --- |
| Control Reference ID | Control Name & Description | Description | Remediation |
| 7.1 | (L2) Ensure HSTS Header is set (Manual) | HTTP Strict Transport Security (HSTS) allows a site to inform the user agent to communicate with the site only over HTTPS. This header takes two parameters: maxage, "specifies the number of seconds, after the reception of the STS header field, during which the user agent regards the host (from whom the message was received) as a Known HSTS Host [speaks only HTTPS]"; and includeSubDomains. includeSubDomains is an optional directive that defines how this policy is applied to subdomains. If includeSubDomains is included in the header, it provides the following definition: this HSTS Policy also applies to any hosts whose domain names are subdomains of the Known HSTS Host's domain name. | Any value greater than 0 meets this recommendation. The examples below are specific to 8 minutes but can be adjusted to meet your requirements. To set the HTTP Header at the server level using an AppCmd.exe command, run the following command from an elevated command prompt: %systemroot%\system32\inetsrv\appcmd.exe set config - section:system.webServer/httpProtocol /+"customHeaders.[name='StrictTransport-Security',value='max-age=480; preload']" To set the HTTP Header and include subdomains at the server level using an AppCmd.exe command, run the following command from an elevated command prompt: %systemroot%\system32\inetsrv\appcmd.exe set config - section:system.webServer/httpProtocol /+"customHeaders.[name='StrictTransport-Security',value='max-age=480; includeSubDomains; preload']" To set the HTTP Header at the Website level using an AppCmd.exe command, run the following command from an elevated command prompt: %systemroot%\system32\inetsrv\appcmd.exe set config "<em>Website"</em> - section:system.webServer/httpProtocol /+"customHeaders.[name='StrictTransport-Security',value='max-age=480; preload']" To set the HTTP Header and include subdomains at the Website level using an AppCmd.exe command, run the following command from an elevated command prompt: %systemroot%\system32\inetsrv\appcmd.exe set config "<em>Website"</em> - section:system.webServer/httpProtocol /+"customHeaders.[name='StrictTransport-Security',value='max-age=480; includeSubDomains; preload']" |
| 7.2 | (L1) Ensure SSLv2 is Disabled (Automated) | The SSLv2 protocol is not considered cryptographically secure, therefore should be disabled. | Perform the following to disable SSL 2.0: 1. Set the following Registry key to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 2.0\Server:Enabled HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 2.0\Client:Enabled 2. Set the following Registry key to 1. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 2.0\Server:DisabledByDefault HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 2.0\Client:DisabledByDefault To disable using PowerShell enter the following command: New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 2.0\Server' -Force | Out-Null New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 2.0\Client' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 2.0\Server' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 2.0\Client' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 2.0\Server' -name 'DisabledByDefault' -value '1' -PropertyType 'DWord' - Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 2.0\Client' -name 'DisabledByDefault' -value '1' -PropertyType 'DWord' - Force | Out-Null |
| 7.3 | (L1) Ensure SSLv3 is Disabled (Automated) | The SSLv3 protocol is not considered cryptographically secure, therefore should be disabled. | Perform the following to disable SSL 3.0: 1. Set the following Registry key to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 3.0\Server:Enabled HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 3.0\Client:Enabled 2. Set the following Registry key to 1. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 3.0\Server:DisabledByDefault HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SS L 3.0\Client:DisabledByDefault To disable using PowerShell enter the following command: New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 3.0\Server' -Force | Out-Null New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 3.0\Client' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 3.0\Server' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 3.0\Client' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 3.0\Server' -name 'DisabledByDefault' -value '1' -PropertyType 'DWord' - Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ SSL 3.0\Client' -name 'DisabledByDefault' -value '1' -PropertyType 'DWord' - Force | Out-Null |
| 7.4 | (L1) Ensure TLS 1.0 is Disabled (Automated) | The TLS 1.0 protocol is not considered cryptographically secure, therefore should be disabled. | Perform the following to disable TLS 1.0: 1. Set the following Registry key to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.0\Server:Enabled HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.0\Client:Enabled 2. Set the following Registry key to 1. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.0\Server:DisabledByDefault HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.0\Client:DisabledByDefault To disable using PowerShell enter the following command: New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.0\Server' -Force | Out-Null New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.0\Client' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.0\Server' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.0\Client' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.0\Server' -name 'DisabledByDefault' -value '1' -PropertyType 'DWord' - Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.0\Client' -name 'DisabledByDefault' -value '1' -PropertyType 'DWord' - Force | Out-Null |
| 7.5 | (L1) Ensure TLS 1.1 is Disabled (Automated) | The TLS 1.1 protocol is not considered cryptographically secure, therefore should be disabled. | Perform the following to disable TLS 1.1: 1. Set the following Registry key to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.1\Server:Enabled HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.1\Client:Enabled 2. Set the following Registry key to 1. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.1\Server:DisabledByDefault HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.1\Client:DisabledByDefault To disable using PowerShell enter the following command: New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.1\Server' -Force | Out-Null New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.1\Client' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.1\Server' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.1\Client' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.1\Server' -name 'DisabledByDefault' -value '1' -PropertyType 'DWord' - Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.1\Client' -name 'DisabledByDefault' -value '1' -PropertyType 'DWord' - Force | Out-Null |
| 7.6 | (L1) Ensure TLS 1.2 is Enabled (Automated) | TLS 1.2 is the most recent and mature protocol for protecting the confidentiality and integrity of HTTP traffic. | Perform the following to enable TLS 1.2: 1. Set the following Registry key to 1. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.2\Server:Enabled 2. Set the following Registry key to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TL S 1.2\Server:DisabledByDefault To enable using PowerShell enter the following command: New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.2\Server' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.2\Server' -name 'Enabled' -value '1' -PropertyType 'DWord' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\ TLS 1.2\Server' -name 'DisabledByDefault' -value '0' -PropertyType 'DWord' - Force | Out-Null |
| 7.7 | (L1) Ensure NULL Cipher Suites is Disabled (Automated) | The NULL cipher does not provide data confidentiality or integrity, therefore it is recommended that the NULL cipher be disabled. | Perform the following to disable NULL cipher: 1. Set the following Registry key to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\NULL :Enabled To disable using PowerShell enter the following command: New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\NU LL' -Force | Out-Null New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\NU LL' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null |
| 7.8 | (L1) Ensure DES Cipher Suites is Disabled (Automated) | The DES Cipher Suite is considered a weak symmetric-key cipher, therefore it is recommended that it be disabled. | Perform the following to disable DES 56/56 cipher: 1. Set the following Registry key to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\DES 56/56:Enabled To disable using PowerShell enter the following command: (Get-Item 'HKLM:\').OpenSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHA NNEL\Ciphers', $true).CreateSubKey('DES 56/56') New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\DE S 56/56' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null |
| 7.9 | (L1) Ensure RC4 Cipher Suites is Disabled (Automated) | The RC4 Cipher Suites are considered insecure, therefore should be disabled. Note: RC4 cipher enabled by default on Server 2012 and 2012 R2 is RC4 128/128. | Perform the following to disable RC4 40/128, RC4 56/128, RC4 64/128, RC4 128/128 ciphers: 1. Set the following Registry keys to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4 40/128:Enabled HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4 56/128:Enabled HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4 64/128:Enabled HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4 128/128:Enabled To disable using PowerShell enter the following commands: (Get-Item 'HKLM:\').OpenSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHA NNEL\Ciphers', $true).CreateSubKey('RC4 40/128') New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC |
| 7.10 | (L1) Ensure AES 128/128 Cipher Suite is Disabled (Automated) | The AES 128/128 Cipher Suite is not considered secure and therefore should be disabled, if possible. | Perform the following to disable AES 128/128 cipher: 1. Set the following Registry key to 0. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AES 128/128:Enabled To disable using PowerShell enter the following command: (Get-Item 'HKLM:\').OpenSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHA NNEL\Ciphers', $true).CreateSubKey('AES 128/128') New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AE S 128/128' -name 'Enabled' -value '0' -PropertyType 'DWord' -Force | Out-Null |
| 7.11 | (L1) Ensure AES 256/256 Cipher Suite is Enabled (Automated) | AES 256/256 is the most recent and mature cipher suite for protecting the confidentiality and integrity of HTTP traffic. Enabling AES 256/256 is recommended. Note: AES 256/256 is enabled by default starting with Server 2012 and 2012 R2. | Perform the following to enable AES 256/256 cipher: 1. Set the following Registry key to 1. HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AES 256/256:Enabled To enable using PowerShell enter the following command: (Get-Item 'HKLM:\').OpenSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHA NNEL\Ciphers', $true).CreateSubKey('AES 256/256') New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AE S 256/256' -name 'Enabled' -value '1' -PropertyType 'DWord' -Force | Out-Null |
| 7.12 | (L2) Ensure TLS Cipher Suite ordering is Configured (Automated) | Cipher suites are a named combination of authentication, encryption, message authentication code, and key exchange algorithms used for the security settings of a network connection using TLS protocol. Clients send a cipher list and a list of ciphers that it supports in order of preference to a server. The server then replies with the cipher suite that it selects from the client cipher suite list. | Perform the following to configure TLS cipher suite order: 1. Set the following Registry key to TLS\_AES\_256\_GCM\_SHA384, TLS\_AES\_128\_GCM\_SHA256,TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_GCM\_SHA384,TLS\_ECDH E\_ECDSA\_WITH\_AES\_128\_GCM\_SHA256,TLS\_ECDHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384,T LS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256,TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CBC\_S HA384,TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CBC\_SHA256,TLS\_ECDHE\_RSA\_WITH\_AES\_25 6\_CBC\_SHA384,TLS\_ECDHE\_RSA\_WITH\_AES\_128\_CBC\_SHA256. HKLM\SOFTWARE\Policies\Microsoft\Cryptography\Configuration\SSL\00010002:Func tions To configure TLS cipher suite order using PowerShell enter the following command: New-Item 'HKLM:\SOFTWARE\Policies\Microsoft\Cryptography\Configuration\SSL\00010002' - Force | Out-Null New-ItemProperty -path 'HKLM:\SOFTWARE\Policies\Microsoft\Cryptography\Configuration\SSL\00010002' - name 'Functions' -value 'TLS\_AES\_256\_GCM\_SHA384,TLS\_AES\_128\_GCM\_SHA256,TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_G CM\_SHA384,TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_GCM\_SHA256,TLS\_ECDHE\_RSA\_WITH\_AES\_256\_ GCM\_SHA384,TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256,TLS\_ECDHE\_ECDSA\_WITH\_AES\_256 \_CBC\_SHA384,TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CBC\_SHA256,TLS\_ECDHE\_RSA\_WITH\_AES\_25 6\_CBC\_SHA384,TLS\_ECDHE\_RSA\_WITH\_AES\_128\_CBC\_SHA256' -PropertyType 'MultiString' -Force | Out-Null |

## 8.3 Ensure Adequate Audit Log Storage

Ensure that logging destinations maintain adequate storage to comply with the enterprise’s audit log management process.  
● ● ●

## 8.5 Collect Detailed Audit Logs

Configure detailed audit logging for enterprise assets containing sensitive data. Include event source, date, username, timestamp, source addresses, destination addresses, and other useful elements that could assist in a forensic investigation.  
● ●

## 8.9 Centralize Audit Logs

Centralize, to the extent possible, audit log collection and retention across enterprise assets.

## 9.1 Associate Active Ports, Services and Protocols to

Asset Inventory Associate active ports, services and protocols to the hardware assets in the asset inventory.  
● ●  
● ●

## 9.2 Ensure Only Approved Ports, Protocols and Services

Are Running Ensure that only network ports, protocols, and services listening on a system with validated business needs, are running on each system.  
● ●  
● ●  
● ●

## 9.5 Implement Application Firewalls

Place application firewalls in front of any critical servers to verify and validate the traffic going to the server. Any unauthorized traffic should be blocked and logged.  
●  
●

## 10.5 Enable Anti-Exploitation Features

Enable anti-exploitation features on enterprise assets and software, where possible, such as Microsoft® Data Execution Prevention (DEP), Windows® Defender Exploit Guard (WDEG), or Apple® System Integrity Protection (SIP) and Gatekeeper™.  
● ●

## 13.10 Perform Application Layer Filtering

Perform application layer filtering. Example implementations include a filtering proxy, application layer firewall, or gateway.

# 14 Controlled Access Based on the Need to Know

Controlled Access Based on the Need to Know

## 14.4 Encrypt All Sensitive Information in Transit

Encrypt all sensitive information in transit.  
● ●  
● ●

## 14.6 Protect Information through Access Control Lists

Protect all information stored on systems with file system, network share, claims, application, or database specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.  
● ● ●

## 16.4 Encrypt or Hash all Authentication Credentials

Encrypt or hash with a salt all authentication credentials when stored.  
● ●  
● ●

## 16.10 Apply Secure Design Principles in Application

Architectures Apply secure design principles in application architectures. Secure design principles include the concept of least privilege and enforcing mediation to validate every operation that the user makes, promoting the concept of "never trust user input." Examples include ensuring that explicit error checking is performed and documented for all input, including for size, data type, and acceptable ranges or formats. Secure design also means minimizing the application infrastructure attack surface, such as turning off unprotected ports and services, removing unnecessary programs and files, and renaming or removing default accounts.

# 18 Application Software Security

Application Software Security