FIPS 140-3 Transition User Guide

Version 1.0

Security Engineering & Architecture Department

PJM Interconnection

Sept. 4, 2024

For Public Use

This page is intentionally left blank.

Contents

Background

PJM will only use cryptographic suites that comply with the National Institute of Standards and Technology (NIST) Federal Information Processing Standards Publication 140-3 (FIPS PUB 140-3) to ensure that its security protocols meet the required North American Energy Standards Board (NAESB) requirements, thereby maintaining the integrity and security of its systems.

Required TLS Settings

PJM will continue to support Transport Layer Security (TLS) versions 1.2 and 1.3.

PJM will continue to support the following cipher suites used in TLS 1.2 and/or 1.3, and PJM customers should make necessary changes to only use these cipher suites.

Cipher Suite ID	Name
0xC02F	ECDHE-RSA-AES128-GCM-SHA256
0xC030	ECDHE-RSA-AES256-GCM-SHA384
0xC027	ECDHE-RSA-AES128-SHA256
0xC028	ECDHE-RSA-AES256-SHA384
0xC02B	ECDHE-ECDSA-AES128-GCM-SHA256
0xC02C	ECDHE-ECDSA-AES256-GCM-SHA384
0xC023	ECDHE-ECDSA-AES128-SHA256
0xC024	ECDHE-ECDSA-AES256-SHA384

Browser Compatibility

Based upon NAESB's guidance, PJM will disable non-FIPS 140-3 compliant ciphers for ExSchedule and OASIS in the PJM Train environment for browser and browserless connections on Sept. 24, 2024. PJM stakeholders can use the Train environment to determine if their browsers are using supported configurations.

The latest versions of web browsers, such as Microsoft Edge, Google Chrome, Mozilla Firefox and Apple Safari, support these FIPS 140-3 compliant ciphers by default. To enable these ciphers in other browsers or ensure they are available, please refer to the respective vendor support documentation.

Guidelines To Determine Browser Protocol and Cipher Suite

Refer to the guidelines below:

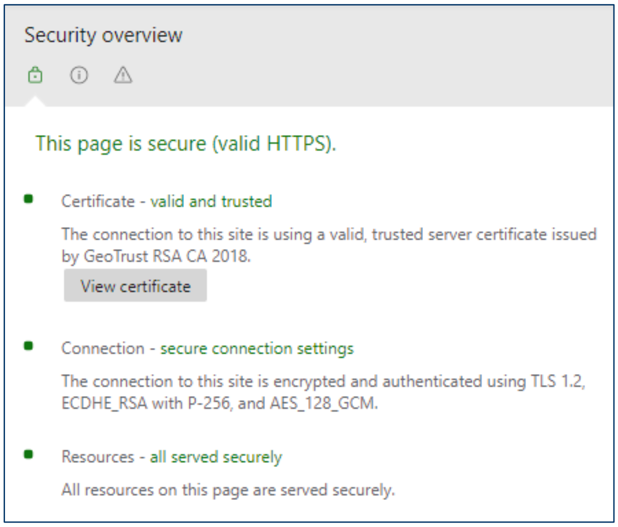
Microsoft Edge

1 | Launch Microsoft Edge.

2 | Enter the URL you want to check in the browser.

3 | Press **CTRL+SHIFT+i**.

4 | Click on the **Security** tab.



Security overview

This page is secure (valid HTTPS).

Certificate - valid and trusted

The connection to this site is using a valid, trusted server certificate issued by GeoTrust RSA CA 2018.

View certificate

Connection - secure connection settings

The connection to this site is encrypted and authenticated using TLS 1.2, ECDHE_RSA with P-256, and AES_128_GCM.

Resources - all served securely

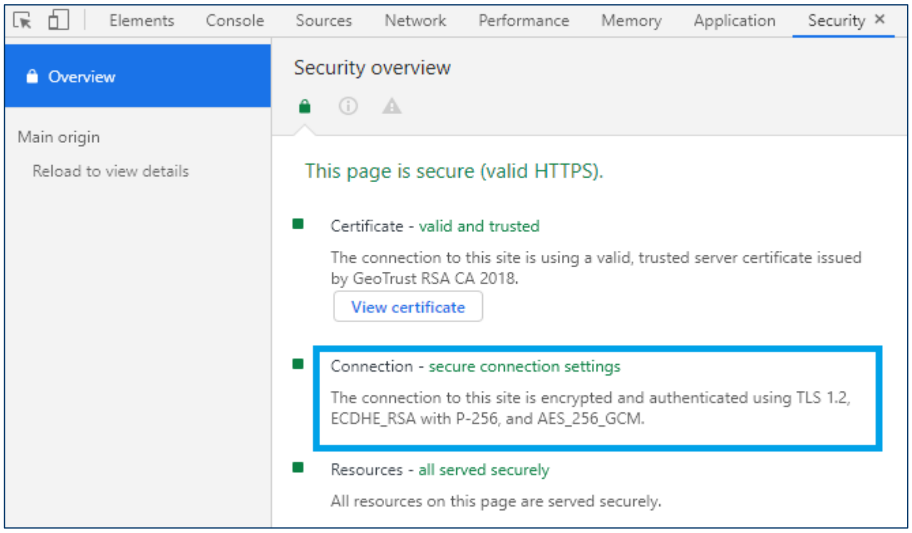
All resources on this page are served securely.

Google Chrome

If you are using Google Chrome version 50 or greater, use the following steps to determine the TLS version and negotiated cipher suite.

1 | Open the Developer Tools with **Crtl+Shift+I** or by clicking on the ⋮ on the Chrome menu > **More tools** > Developer tools, and then click on the **Security** tab.

2 | The information has now been added to the 🔒 **Overview** section without reloading and includes the key exchange group.



Security overview

This page is secure (valid HTTPS).

Certificate - valid and trusted

The connection to this site is using a valid, trusted server certificate issued by GeoTrust RSA CA 2018.

View certificate

Connection - secure connection settings

The connection to this site is encrypted and authenticated using TLS 1.2, ECDHE_RSA with P-256, and AES_256_GCM.

Resources - all served securely

All resources on this page are served securely.

Mozilla Firefox

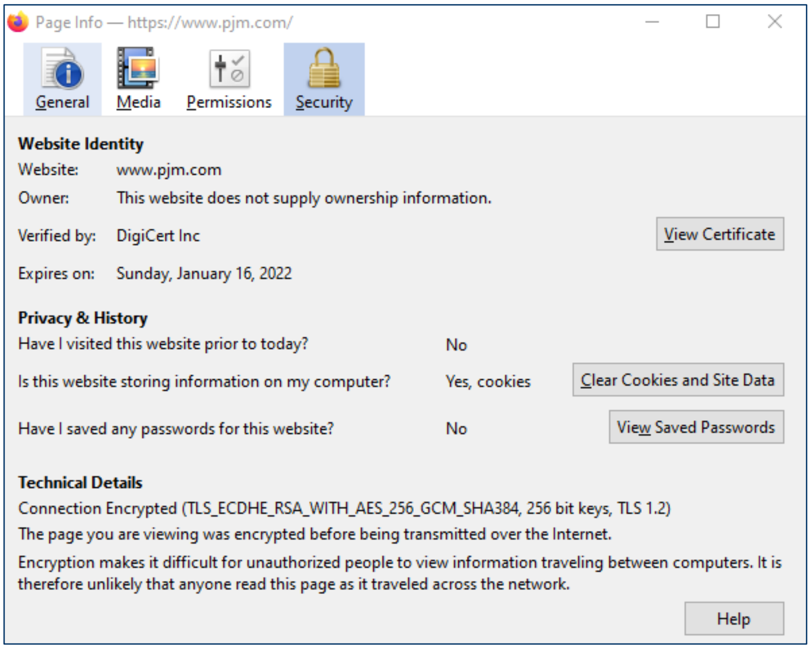
1 | Open Firefox.

2 | Enter the URL you want to check in the browser.

3 | Click on the 🔒 lock icon in the location bar.

4 | Click on the arrow next to connection secure, and then click on **More Information**.

5 | In the new window, look for the **Technical Details** section.



Website Identity
Website: www.pjm.com
Owner: This website does not supply ownership information.
Verified by: DigiCert Inc
Expires on: Sunday, January 16, 2022

Privacy & History
Have I visited this website prior to today? No
Is this website storing information on my computer? Yes, cookies
Have I saved any passwords for this website? No

Technical Details
Connection Encrypted (TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384, 256 bit keys, TLS 1.2)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.

Guidelines To Configure Browserless APIs To Use TLS 1.2

Refer to the guidelines below:

Java (Oracle)

Java 8 (1.8) and higher	Compatible with TLS 1.2 by default
Java 7 (1.7) and higher, Java 6u121 and higher	Enable TLS 1.2 using the https.protocols Java system property for HttpsURLConnection. To enable TLS 1.2 on non-HttpsURLConnection connections, set the enabled protocols on the created SSLSocket and SSLEngine instances within the application source code. It's recommended to test the change before deploying to production servers.
Below Java 6u121	Not compatible with TLS 1.2

Java (IBM)

Java 8	Compatible with TLS 1.2 or higher by default. You may need to set com.ibm.jsse2.overrideDefaultTLS=true if your application or a library with this name uses SSLContext.getinstance (TLS).
Java 7 and higher, Java 6.0.1 service refresh 1 (J9 VM2.6) and higher, Java 6 service refresh 10 and higher	Enable TLS 1.2 using the https.protocols Java system property for HttpsURLConnection and the com.ibm.jsse2.overrideDefaultProtocol Java system property for SSLSocket and SSLEngine connections. You may also need to set com.ibm.jsse2.overrideDefaultTLS=true. It's recommended to test the change before deploying to production servers.

.NET

.NET 4.6 and higher	Compatible with TLS 1.2 by default
.NET 4.5 to 4.5.2	.NET 4.5, 4.5.1 and 4.5.2 do not enable TLS 1.2 by default. Two options exist to enable these, as described below. **Option 1:** .NET applications may directly enable TLS 1.2 in their software code by setting System.Net.ServicePointManager.SecurityProtocol to enable SecurityProtocolType.Tls12 and SecurityProtocolType.Tls11. The following C# code is an example: System.Net.ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12 | SecurityProtocolType.Tls11 | SecurityProtocolType.Tls; **Option 2:** It may be possible to enable TLS 1.2 by default without modifying the source code by setting the SchUseStrongCrypto DWORD value in the following two registry keys to 1, creating them if they don't exist: "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\.NETFramework\v4.0.30319" and "HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\.NETFramework\v4.0.30319" Although the version number in the registry keys are 4.0.30319, the .NET 4.5, 4.5.1 and 4.5.2 frameworks also use these values. Those registry keys, however, will enable TLS 1.2 by default in all installed .NET 4.0, 4.5, 4.5.1 and 4.5.2 applications on that system. It is advisable to test this change before deploying it to your production servers. These registry values, however, will not affect .NET applications that set the System.Net.ServicePointManager.SecurityProtocol value.
.NET 4.0	.NET 4.0 does not enable TLS 1.2 by default. To enable TLS 1.2 by default, it is possible to install .NET Framework 4.5, or a newer version, and set the SchUseStrongCrypto DWORD value in the following two registry keys to 1, creating them if they don't exist: "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\.NETFramework\v4.0.30319" and "HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\.NETFramework\v4.0.30319" Those registry keys, however, may enable TLS 1.2 by default in all installed .NET 4.0, 4.5, 4.5.1 and 4.5.2 applications on that system. We recommend testing this change before deploying it to your production servers. These registry values, however, will not affect .NET applications that set the System.Net.ServicePointManager.SecurityProtocol value.
.NET 3.5, .NET 3.0 and .NET 2.0	We recommend upgrading to .NET 4.6 or higher. If you cannot upgrade, to use TLS 1.2 in .NET 2.0, .NET 3.0 and NET 3.5, the value 3072 needs to be set in the security protocol. It's recommended to test the change before deploying to production servers.

Guidelines To Configure Browserless APIs

Refer to the guidelines below:

Guidelines When Using HTTPS Proxy Server

Some networks intercept outbound HTTPS traffic by using a proxy server that creates its own certificates, so that the unencrypted communications with PJM and other websites can be inspected. Those proxy servers create their own TLS connections to PJM websites. Networks that use this type of proxy server need to ensure that they support TLS 1.2 and prefer TLS 1.2 when connecting to PJM websites. Irregular behavior may be observed if the proxy server either does not support TLS 1.2 or prefers TLS 1.1 over TLS 1.2 when connecting to PJM websites.

The general configuration recommendations for intercepting HTTPS proxy servers regarding the TLS 1.1 disablement are the following:

- If HTTPS interception is required by the company’s policy, or otherwise cannot be removed or exempted, update that proxy server to a newer version that supports TLS 1.2.
- If the intercepting HTTPS proxy server does support TLS 1.2, but prefers TLS 1.1 by using it in its initial Client Hello messages, update the proxy server’s configuration to prefer TLS 1.2 over TLS 1.1 when connecting to PJM websites *.pjm.com.

Common Errors When Using Unsupported TLS Configuration

When a noncompliant source device tries to connect to a PJM application that is FIPS 140-3 compliant, the following errors are observed at the source device.

Browsers	ERR_SSL_VERSION_OR_CIPHER_MISMATCH
Browserless Java API	javax.net.ssl.SSLHandshakeException: Received fatal alert: handshake_failure
Browserless .NET API	NET API system.Net.Http.HttpRequestException: The SSL connection could not be established.