TLS 1.3 Transition User Guide

Security Engineering & Architecture Department

PJM Interconnection

June 4, 2026

For Public Use

This page is intentionally left blank.

Contents

Background

PJM will transition its external-facing web applications and APIs to Transport Layer Security (TLS) version 1.3 while continuing to permit coexistence with TLS 1.2 for a defined transition period.

This approach ensures stakeholders have sufficient time to modernize client software while maintaining compliance with:

- NIST Federal Information Processing Standards Publication 140-3 (FIPS PUB 140-3)
- North American Energy Standards Board (NAESB) cybersecurity standards

TLS 1.3 simplifies protocol negotiation, removes obsolete cryptographic primitives and improves connection performance. The transition to TLS 1.3 will also enable a clear pathway for implementing Quantum Safe Cryptography on all PJM systems, further enhancing the confidentiality of encrypted communications.

Over time, PJM intends for TLS 1.3 to become the preferred and predominant protocol, with TLS 1.2 support eventually retired following industry guidance and stakeholder readiness.

PJM is providing this document to aid PJM stakeholders in the transition to TLS 1.3. PJM will provide assistance to stakeholders seeking to understand or clarify details in this document. However, due to the varied browser and browserless environments that PJM stakeholders use, PJM is unable to provide additional technical support.

Required TLS Settings

PJM will support both TLS 1.2 and TLS 1.3 during the transition period.

- TLS 1.3 is preferred for all new integrations and upgrades.
- TLS 1.2 remains permitted to ensure backward compatibility with legacy systems.
- Protocol negotiation will be server-driven, selecting the highest mutually supported version.

Stakeholders are strongly encouraged to enable TLS 1.3 and validate behavior in the PJM Train environment. Where possible, TLS 1.3 should be preferred over TLS 1.2.

Supported Cipher Suites and Key Exchanges

TLS 1.3 Cipher Suites (Preferred)

TLS 1.3 uses a simplified cipher suite structure and always provides authenticated encryption, ephemeral key exchange and forward secrecy.

The following TLS 1.3 cipher suites are supported by PJM:

Cipher Suite ID	Name
0x1301	TLS_AES_128_GCM_SHA256
0x1302	TLS_AES_256_GCM_SHA384
0x1303	TLS_CHACHA20_POLY1305_SHA256

TLS 1.2 Cipher Suites (Permitted During Coexistence)

During the coexistence period, PJM will continue to support a restricted set of secure TLS 1.2 cipher suites, including:

Cipher Suite ID	Name
0xC02F	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
0xC030	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

Browser Compatibility

During the coexistence period, PJM environments will negotiate either TLS 1.3 or TLS 1.2 depending on client capabilities. Many modern browsers have supported TLS 1.3 since late 2018.

The following browsers have TLS 1.3 capabilities enabled by default:

Microsoft Edge (Chromium-based)	**Google Chrome**	**Mozilla Firefox**	**Apple Safari**

Stakeholders are encouraged to verify that TLS 1.3 is successfully negotiated when connecting to PJM Train environments. To enable support for TLS 1.3 in other browsers or ensure it is available, please refer to the respective vendor support documentation.

Guidelines To Determine Browser Protocol and Cipher Suite

Refer to the guidelines below:

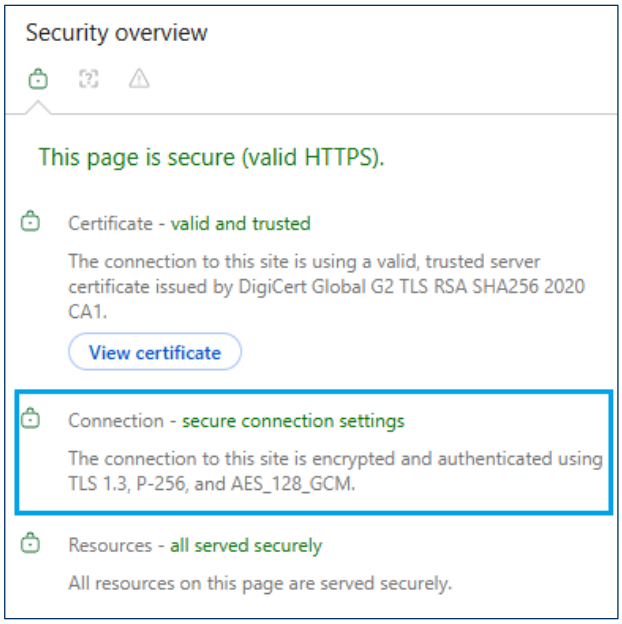
Microsoft Edge

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.

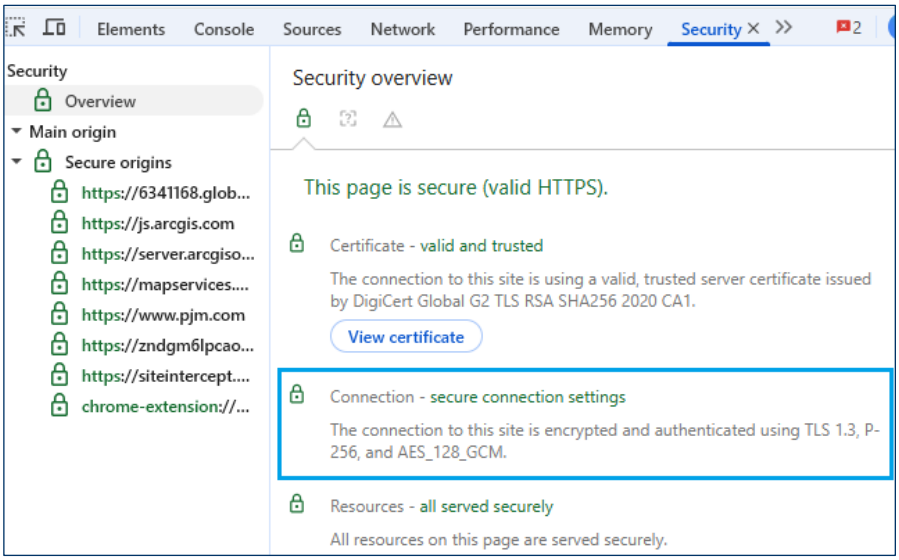


Google Chrome

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.



Mozilla Firefox

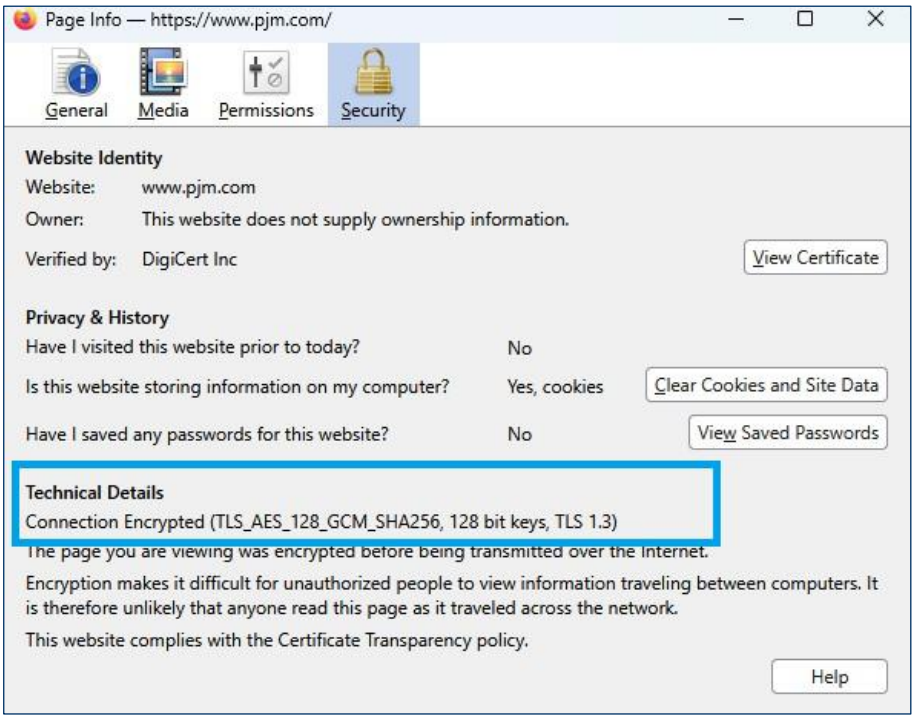
1 | Open Firefox.

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.



Guidelines To Configure Browserless APIs To Use TLS 1.3

Refer to the guidelines below:

Java (Oracle)

Java Version	TLS 1.3 Support
Java 11+	TLS 1.3 enabled by default
Java 8u261+	TLS 1.3 supported; may require explicit enablement.
Below Java 8u261	Not compatible with TLS 1.3

To prefer TLS 1.3 where available:

```
-Djdk.tls.client.protocols=TLSv1.3,TLSv1.2
```

Applications using SSLSocket or SSLEngine should allow protocol negotiation and avoid hard-coding TLS 1.3 only.

Java (IBM)

Java Version	TLS 1.3 Support
Java 11+	TLS 1.3 enabled by default
Java 8 Service Refresh 6, Fix Pack 25+	TLS 1.3 supported; may require explicit enablement.
Below Java 8 Service Refresh 6, Fix Pack 25	Not compatible with TLS 1.3

.NET

.NET Version	TLS 1.3 Support
.NET 5.0+ on modern OS	TLS 1.3 supported
.NET Framework 4.8+	TLS 1.3 supported

TLS 1.3 support depends on both the .NET runtime and the operating system. Ensure that the Operating System is configured to enable TLS 1.3 connections. Applications should allow OS-level protocol negotiation and avoid explicitly restricting protocol versions.

Refer to Microsoft's documentation for more information on enabling TLS 1.3 in .NET and .NET Framework:

- TLS/SSL best practices – .NET | Microsoft Learn
- Transport Layer Security (TLS) best practices with .NET Framework | Microsoft Learn

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 1.3 and prefer TLS 1.3 when connecting to PJM websites.

The general configuration recommendations for intercepting HTTPS proxy servers regarding the TLS 1.2 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 1.3.
- If the intercepting HTTPS proxy server does support TLS 1.3, but prefers TLS 1.2 by using it in its initial Client Hello messages, update the proxy server's configuration to prefer TLS 1.3 over TLS 1.2 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 has been configured to accept TLS 1.2 and TLS 1.3 connections, the following errors are observed at the source device.

Software	Error message
Browsers	`ERR_SSL_VERSION_OR_CIPHER_MISMATCH`
Browserless Java API	`SSLHandshakeException: handshake_failure`
Browserless .NET API	`HttpRequestException: The SSL connection could not be established`

Post-Quantum and Hybrid Key Exchange Considerations

PJM environments may negotiate hybrid key exchange mechanisms in TLS 1.3 to provide protection against future cryptanalytic advances, including quantum-capable adversaries. PJM will prefer these hybrid key exchange mechanisms and may require them in the future based on industry standards and vendor readiness.

When supported by both client and server, TLS 1.3 connections may derive session keys using a hybrid construction (X25519MLKEM768) combining:

- X25519 (elliptic-curve Diffie-Hellman)
- ML-KEM-768 (a NIST-standardized lattice-based key encapsulation mechanism)

In this hybrid model:

- Both classical and post-quantum shared secrets are generated.
- The final session key is derived by cryptographically combining both secrets.
- Security is maintained as long as either algorithm remains secure.

This approach is transparent to applications but may affect TLS handshake size, proxy server compatibility and legacy TLS inspection tools.

Stakeholders should ensure that TLS libraries, proxies and security appliances in their environments support TLS 1.3 hybrid key exchange, specifically X25519MLKEM768.

Software	Minimum Version for Hybrid Key Exchange Support
.NET	.NET 10
Java	OpenJDK 27 (unreleased, GA targeted for 14 September 2026)
Chrome	Chrome 131
Edge	Edge 131
Firefox	Firefox 132

A transition from hybrid key exchange to ML-KEM-only key exchange would be considered only after post-quantum algorithms and supporting technologies demonstrate sufficient long-term maturity, interoperability and operational stability.