SSL/TLS

SSL/TLS – OVERVIEW

- **What does SSL stand for?**
  - Secure Socket Layer
  - A socket is the technical term for a network connection between machines
  - SSL is a layer between the TCP/IP interface and your application

- **TLS is the new terminology**
  - Transport Layer Security
  - Removes the reference to “Socket”
  - Can (in theory) be used on any communication

SSL/TLS – OVERVIEW

PxPlus will connect with SSL 2.0 and above

This can be controlled

Protocol	Published	Status
SSL 1.0	Unpublished	Unpublished
SSL 2.0	1995	Deprecated in 2011
SSL 3.0	1996	Deprecated in 2015
TLS 1.0	1999	Deprecated in 2021
TLS 1.1	2006	Deprecated in 2021
TLS 1.2	2008	In use since 2008
TLS 1.3	2018	In use since 2018

SSL/TLS - OVERVIEW

- **SSL/TLS provides three main services**
  - Data encryption
  - Authentication of the server to the client
  - Authentication of the client to the server (less common)

- **PxPlus uses the industry standard OpenSSL to provide SSL/TLS support**
  - On Windows, PxPlus ships with OpenSSL included
    - Version is updated with major PxPlus version (Windows)
  - On UNIX/Linux, OpenSSL is part of the OS
  - Used by [TCP], Simple Client-Server, Email, EZWeb, ODBC, PxServer
  - Specify which OpenSSL PxPlus should use by setting the environment variables **PXP_CRYPTO_LIB** and **PXP_SSL_LIB**
  - It is also possible to query which version of OpenSSL PxPlus is using by issuing a **TCB**("OpenSSL_Version")

SSL/TLS - DATA ENCRYPTION

Ciphers provide “reversible” encryption

- Data encrypted by “**Encryption key**” can only be decrypted by “**Decryption key**”
  - Key size and the algorithm determines how secure data is
  - Typical key sizes range from 128 to 4096 bits
    - 32 bit is over 4 billion thus 4096 is quite large
  - Algorithms can be found to be faulty and “leak” answers

No cipher is 100% safe - all can be cracked given enough resources and time

SSL/TLS – DATA ENCRYPTION

SSL/TLS encryption algorithms (ciphers)

Only use modern and unbroken ciphers

Method	Description
aes	**Advanced Encryption Standard** (AES), also known as Rijndael, adopted as an encryption standard by the US government.
aria	**ARIA** is a block cipher with a block size of 128 bits and key sizes of 128, 192 and 256 bits. It was designed in 2003 by a large group of South Korean researchers. In 2004, the Korean Agency for Technology and Standards selected it as a standard cryptographic technique.
camellia	**Camellia** is a symmetric key block cipher with a block size of 128 bits and key sizes of 128, 192 and 256 bits. It was jointly developed by Mitsubishi Electric and NTT of Japan. The cipher has been approved for use by the ISO/IEC, the European Union's NESSIE project and the Japanese CRYPTREC project. The cipher has security levels and processing abilities comparable to the Advanced Encryption Standard.
chacha20	**ChaCha20** is a stream cipher developed by Daniel J. Bernstein. It was designed in 2005 and then later submitted to the eSTREAM European Union cryptographic validation process by Bernstein.
sm4	**SM4** (formerly SMS4) is a block cipher used in the Chinese National Standard for Wireless LAN WAPI (WLAN Authentication and Privacy Infrastructure).

SSL/TLS - DATA ENCRYPTION

How are keys used?

To send data securely to the host

- Encryption key is made **PUBLIC**
  - Key is used to encrypt data
  - Based on the **PRIVATE** key
- Decryption key is kept **PRIVATE** on host
  - **Never** should be revealed

SSL/TLS - DATA ENCRYPTION

Which cipher is used?

- Server and client negotiate which ciphers they support
  - Client identifies which ciphers it supports
    - Supplied in order of preference
  - Server identifies which cipher it wants
  - Client confirms
- Server will reject any it considers unsafe or unsupported
  - Connection fails if none are acceptable

Using insecure ciphers will result in PCI Compliance failure

SSL/TLS - AUTHENTICATION

Validation/Authentication of system done using certificates (X509)

- Certificate contains the following:
  - Server Name/Address
  - Start/End Dates for which certificate is valid
  - Issuer identification
    - Name, Country, City, State/Province
  - Public key
- Certificates exchanged during negotiation
- <u>**SHOULD**</u> be validated for secure connection

SSL/TLS - AUTHENTICATION

What is generally validated?

- Current date is within the start and end dates for certificate
- The server address on the certificate matches the server we connected to
- The certificate was issued by a trusted certificate authority
  - The certificate can be found in a list of trusted certificates

Optional test

- Match to previously known Public key

SSL/TLS - AUTHENTICATION

Normally, only Server provides certificate

- Client only provides a Public key

When would Client require certificate?

- Controlled access to specific pre-cleared clients
  - Cannot connect unless you have a known certificate

SSL/TLS - HOW TO ESTABLISH TRUST

SSL/TLS provides a mechanism that establishes "TRUST"

- There are KNOWN "Trusted" companies that provide "certificates"
    - Known as "Certificate Authorities" (**CA**)
    - Most commonly used CA are:
        - Let's Encrypt
        - GlobalSIgn
        - IdenTrust
        - Sectigo (Comodo Cybersecurity)
        - DigiCert
        - GoDaddy

SSL/TLS - GETTING A “TRUSTED” CERTIFICATE

You need a certificate from a CA for HTTPS

- If not trusted, browsers will complain
    - Expired certificate is the most common
    - Most will reject connection
    - Certificate **MUST** match site name

How to obtain a certificate?

- 1 year certificate
    - Contact a CA provider
    - Costs around $100+ per year
    - Requires company background check
- 90-day certificate
    - Let’s Encrypt
    - Free
    - Auto renews via domain validation

SSL/TLS – "SELF-SIGNED" CERTIFICATE

You can generate a certificate for yourself

- By default, it will not be trusted
- Can be used by all SSL/TLS software
  - Application can decide if TRUST is required
  - If TRUST required, users can add it to their local store
  - Includes all the same data as a standard certificate

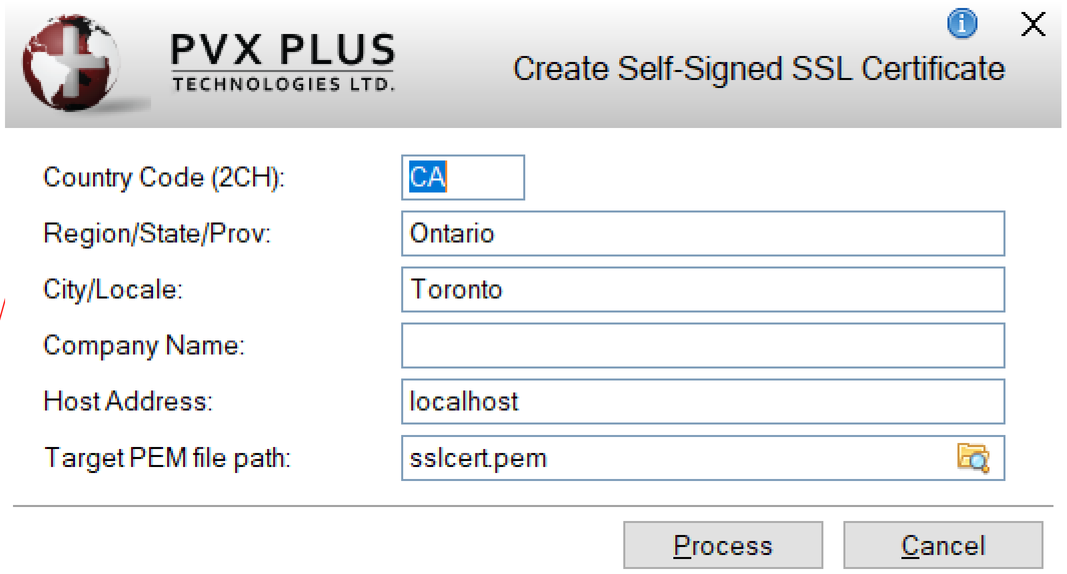
SSL/TLS – "SELF-SIGNED" CERTIFICATE

PxPlus includes *TOOLS/SSLCERT utility to create self-signed certificate

- **To generate a file:**

  run "*tools/sslcert"

- Uses Internet to create certificate on our servers
- Returns single PEM file with certificate and key information
- Generates 2048-bit key
- Text mode version also available



Country Code (2CH):	CA
Region/State/Prov:	Ontario
City/Locale:	Toronto
Company Name:	
Host Address:	localhost
Target PEM file path:	sslcert.pem

SSL/TLS - PXPLUS SSL OPTIONS

- **To make a client connection using SSL/TLS use the [TCP] option SECURE**

```
OPEN (HFN) "[TCP]https://myserver.com/app;443;SECURE"
```

- **To make a server using SSL/TLS use the [TCP] SECURE=xxxx option**
  - Where xxxx is the path to the certificate
  - Supports X509 certificates created for use with OpenSSL or Apache

SSL/TLS - PXPLUS SSL OPTIONS

- **An X509 certificate can be in the form of a PEM file, which contains both the certificate and Private key**
  - The [TCP] option **SECURE**=xxx can be used to specify the certificate file

```
OPEN (HFN) "[TCP];443;SECURE=/etc/certs/mycert.pem"
```

- **An X509 certificate can be in the form of two PEM files, one containing the certificate and the other containing the Private key**
  - The [TCP] option **SECURE**=xxx can be used to specify the certificate file while **PRIVKEY=***xxx* can be used to specify the private key file

```
OPEN (HFN) "[TCP];443;SECURE=/etc/letsencrypt/live/exp.com/fullchain.pem;PRIVKEY=/etc/letsencrypt/live/exp.com/privkey.pem"
```

- **You may get a certificate in a different format, such as the Microsoft PFX file format**
  - Convert to a PEM file using the PxPlus utility, *TOOLS/PFXCERTCONVERT

```
CALL "*tools/pfxcertconvert", "C:\ProgramData\Certify\certes\assets\pfx\exp.com.pfx", "password", "converted.pem"
OPEN (HFN) "[TCP];443;SECURE=converted.pem"
```

SSL/TLS - PXPLUS SSL OPTIONS

Defining Supported Protocol

To suppress any of these protocols:

NoSSLv2, NoSSLv3, NoTLSv1, NoTLSv1.1, NoTLSv1.2, NoTLSv1.3

To force one specific protocol:

TLS, TLS1.1, TLS1.2, TLS1.3

- Default will connect using any protocol from SSL v2 through TLS 1.3

```
OPEN (HFN) "[TCP];443;SECURE=/etc/certs/mycert.pem;TLSv1.3"
```

SSL/TLS - PXPLUS SSL OPTIONS

Certificate Validation

Certificates= Ignore | Validate | Trust

Default set using PVX_CERTIFICATES environment variable

- **Ignore** doesn't validate certificate (default)
- **Validate** makes sure certificate:
  - Is not expired
  - Is for the proper server by matching name
- **Trust** extends Validation
  - Certificate must have come from trusted CA
  - PxPlus ships with list of trusted certificates

    ***<pxplus exe directory>*/ca-bundle.crt**
  - This file **MUST** be updated periodically

Can be changed using PVX_CERTSTORE environment variable

https://raw.githubusercontent.com/bagder/ca-bundle/master/ca-bundle.crt

```
OPEN (HFN) "[TCP];443;SECURE;certificates=Validate"
```

SSL/TLS - PXPLUS SSL OPTIONS

Defining Acceptable/Supported Ciphers

Ciphers= *list of accepted ciphers*

- Contents of list defined at www.openssl.org
- PCI compliance **(currently)**

Ciphers=ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256

- Includes only known strong ciphers

```
OPEN (HFN) "[TCP];443;SECURE=/etc/certs/mycert.pem;Ciphers=ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256;NoSSLv2;NoSSLv3;NoTLSv1;NoTLSv1.1"
```

SSL/TLS - EZWEB

- **Specify an SSL certificate when launching EZWeb server to enable SSL/TLS encryption**
  - EZWeb supports X509 combined and separated PEM files

```
/app/pxplus "*ezweb/server" -arg 443 "/etc/certs/mycert.pem"
```

```
/app/pxplus "*ezweb/server" -arg 443 "/etc/letsencrypt/live/exp.com/fullchain.pem
privkey=/etc/letsencrypt/live/exp.com/privkey.pem"
```

  - EZWeb supports PFX certificates without conversion
    - If the PFX is password protected, use **pfxpswd**= option

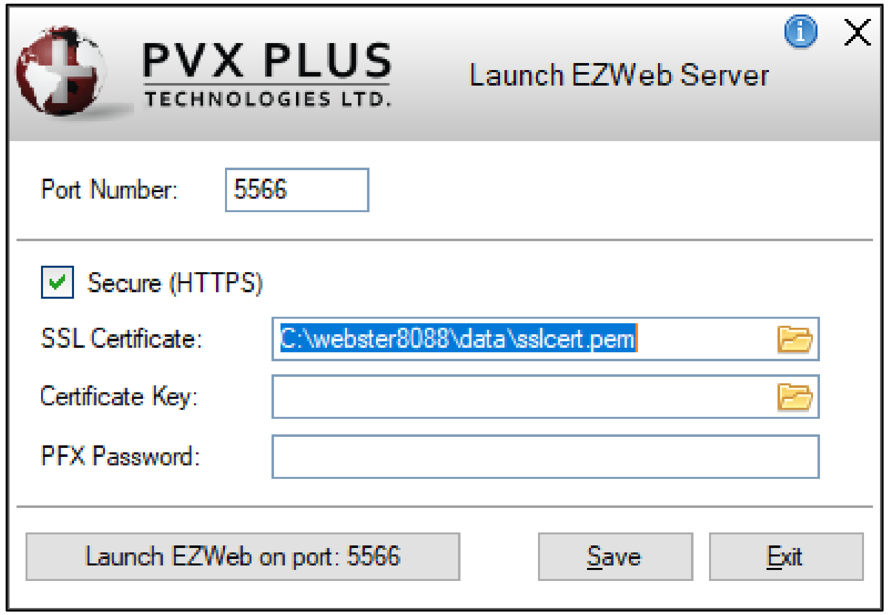
```
"C:\app\pxplus.exe" *ezweb\server -arg 443 "C:\ProgramData\Certify\certes\assets\pfx\exp.com.pfx pfxpswd=password"
```

  - Also can be specified via **ezweb.conf** file
    - **SECURE** - to specify combined PEM file, certificate PEM file or PFX file
    - **PRIVKEY** - to specify Private key PEM file
    - **PFXPSWD** - to specify password for the PFX file

```
port 443
secure "/etc/letsencrypt/live/exp.com/fullchain.pem"
privkey "/etc/letsencrypt/live/exp.com/privkey.pem"
nobrowse
```

SSL/TLS - EZWEB

- **Can launch EZWeb Server using graphical utility**
  - IDE >Web Deployment > Launch EZWeb Server
  - Supports same security options as command line
- **System Tray messages updated when secure**



PVX PLUS TECHNOLOGIES LTD. Launch EZWeb Server

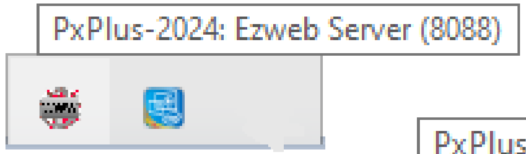
Port Number: 5566

[✓] Secure (HTTPS)

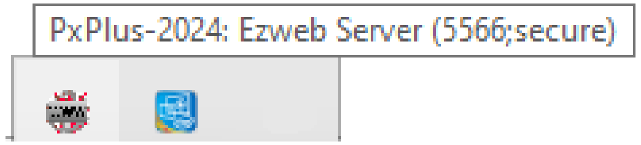
SSL Certificate: C:\webster8088\data\sslcert.pem

Certificate Key:

PFX Password:



PxPlus-2024: Ezweb Server (8088)



PxPlus-2024: Ezweb Server (5566;secure)

SSL/TLS - PXPLUS CLIENT-SERVER AND SSL

- **When the CS host/client is launched, [TCP] options can be used to specify SSL/TLS options**

```
pxplus.exe *plus\cs\host -arg 12345;secure= C:\app\certs\app.pem;TLS1.3
```

```
pxplus.exe *plus\cs\client -arg MySrvr;12345;secure;certificates=Validate
```

Host-side CS options (server)

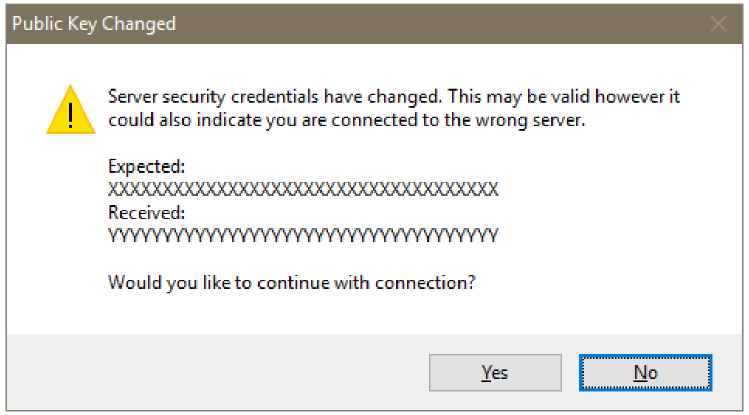
Default options can be set in:
PXP_CS_OPT
Environment variable

Client-side CS options (workstation)

Default options can be set in:
PXP_CS_OPT_CLIENT
Environment variable

SSL/TLS - PXPLUS CLIENT-SERVER AND SSL

- **PUBKEY=xxxxxxx can be used on the client to specify which Public key to accept from the server or to ask the user to confirm**
  - If *xxxxxxx* contains the word "**check**", then on the first connect to the server, the client process will ask the user to confirm that the Public key signature it received is correct



Public Key Changed

Server security credentials have changed. This may be valid however it could also indicate you are connected to the wrong server.

Expected:
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Received:
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY

Would you like to continue with connection?

  - If *xxxxxxx* contains a Public key signature (Base 64 of the SHA-256 of the X509 Public key), then it **MUST** match the server value

```
pxplus.exe *plus\cs\client -arg MySrvr;12345;secure;pubkey=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

SSL/TLS - FUTURE CONSIDERATIONS

SSL is constantly changing to address new vulnerabilities

- Maintain your PxPlus version current
  - We update SSL to latest options with each release
- On Linux, keep your OpenSSL current
- For Windows, we ship current OpenSSL libraries
- If using **trust** relationships, update **ca-bundle.crt**