OpenADR 2.0 Security

Jim Zuber, CTO
QualityLogic, Inc.
Security Overview

- Client and server x.509v3 certificates
- TLS 1.2 with SHA256 ECC or RSA cipher suites
  - TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
  - TLS_RSA_WITH_AES_128_CBC_SHA256
- Optional XML payload signatures
- Requirements above are “out of box”, deployment security may differ
Focus

- Most common support issue for OpenADR implementers is getting communication going over a secure channel
- This presentation will focus on the minimum necessary concepts and steps required to be successful setting up secured OpenADR communication
Cipher suites

- An algorithm for performing encryption or decryption.

- OpenADR support two types of cipher suites
  - ECC (Elliptical Curve Cryptography)
  - RSA

- The OpenADR RSA and ECC cipher suites used by OpenADR have a strong “SHA256” hash algorithm
X.509 Certificates

- **Root Certificate**
  - Used to validate the identity of a counterparty such as a VEN trying to talk to a VTN
  - Certificates are managed by a Certificate Authority (CA), NetworkFX for OpenADR
  - OpenADR has a public cert each for RSA and ECC

- **Intermediate Root Certificate**
  - Signed by the root cert’s private key
  - Used in conjunction with root cert for identifying counterparties
  - VENs and VTNs have separate intermediate certs
X-509 Certificates

- **Device Certificate**
  - Identifies a specific VTN or VEN
  - Signed by the intermediate cert’s private key

- **Private Key**
  - Matching device certificate private key.
  - Used to encrypt hash of traffic payloads, which can only be decrypted with public device cert
Cert Tree

NetworkFX Certificate Authority

- RSA Public Cert
  - VEN RSA Intermediate Cert
    - VEN RSA Device Cert
    - VEN RSA Private Key
  - VTN RSA Intermediate Cert
    - VTN RSA Device Cert
    - VTN RSA Private Key

- ECC Public Cert
  - VEN ECC Intermediate Cert
    - VEN ECC Device Cert
    - VEN ECC Private Key
  - VTN ECC Intermediate Cert
    - VTN ECC Device Cert
    - VTN ECC Private Key
TLS Negotiation

- **TLS** is a cryptographic protocol designed to provide communications security over a computer network. Think “HTTPS”.
- **TLS negotiation** does the following
  - Sets the TLS version to use (TLS 1.2 for OpenADR)
  - Sets the cipher suites to use
  - Exchanges client and server device certificates
  - Validates certificates are trusted
  - Establishes symmetric encryption key for data exchange
TLS Negotiation

VTN (ISO or Utility)

Client Hello (TLS/Cipher Options) →

Server Hello (selected TLS/Cipher) →

Send Server Device Certificate →

Request Client Certificate →

Send Client Device Certificate →

Establish Symmetric Encryption Key →

VEN (C&I, SMB)
Certificate and Stores

- Certificates are available in a number of formats:
  - **PEM** Format – Base64 encoded certificate
  - **DER** Format – Binary encoded certificates
  - **PKCS12** Format – Contains both public device and private key

- Stores
  - **Trust Store** – Contains public root and intermediate Certs from counterparty
  - **Key Store** – Contains device cert and private key
Certificate Management Tools

- **OpenSSL** and **Keytool**
- Convert between certificate formats
- Create trust and key stores
- Verify device certs validate against public certs (OpenSSL)
Test Certificates

- `ww.networkfx.net`

DNS name is stores in cert CN field. Must match VTN URL if host authentication not disabled.
The MAC address is stored in the CN field of the VEN certificate without colons. If using OpenFire XMPP Server, the XMPP user name must match the VEN CN Name in the Certificate.
## HTTP Certificate Configuration Example

<table>
<thead>
<tr>
<th></th>
<th>VEN</th>
<th>VTN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Trust Store</strong></td>
<td>Common Root Cert</td>
<td>Common Root Cert</td>
</tr>
<tr>
<td></td>
<td>VTN Intermediate Cert</td>
<td>VEN Intermediate Cert</td>
</tr>
<tr>
<td><strong>Key Store</strong></td>
<td>VEN Device Cert</td>
<td>VTN Device Cert</td>
</tr>
<tr>
<td></td>
<td>VEN Private Key</td>
<td>VTN Private Key</td>
</tr>
</tbody>
</table>

All Certs must be of a common type: RSA or ECC. Can be both
XMPP Transport

All communication flows through the XMPP Server

Note that the connection between the XMPP server and the VTN is a private connection. From the VEN’s perspective the XMPP server “is” the VTN.
## XMPP Certificate Configuration Example

<table>
<thead>
<tr>
<th></th>
<th>VEN</th>
<th>XMPP Server</th>
<th>VTN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Trust Store</strong></td>
<td>Common Root Cert</td>
<td>Common Root Cert</td>
<td>Common Root Cert</td>
</tr>
<tr>
<td></td>
<td><strong>VTN Intermediate Cert</strong></td>
<td><strong>VEN Intermediate Cert</strong></td>
<td><strong>VTN Intermediate Cert</strong></td>
</tr>
<tr>
<td><strong>Key Store</strong></td>
<td><strong>VEN Device Cert</strong></td>
<td><strong>VTN Device Cert</strong></td>
<td><strong>VEN Device Cert</strong></td>
</tr>
<tr>
<td></td>
<td><strong>VEN Private Key</strong></td>
<td><strong>VTN Private Key</strong></td>
<td><strong>VEN Private Key</strong></td>
</tr>
</tbody>
</table>

The connection between the VTN and the XMPP server is a private connection. The cert configuration shown for the VTN is just a suggestion to deal with the OpenFire’s CN name to user name match requirement.
Fingerprint

- The VEN Zip package with certificates from NetworkFX will contain a fingerprint file.
- This fingerprint value may need to be installed in the VTNs back-end server configuration in order to interoperate.
Questions?