Smart Meter Texas
Web Services Security Interface Document

Prepared for

Texas Competitive Electric Market

AMIT Working Group

November 20, 2014

Table of Contents

1 Introduction 3
1.1 Prerequisite for Interfaces 3
2 SMT WS-Security Implementation features 4
3 Trust relationship Establishment 5
3.1 Trust relationships in Production 5
3.2 Trust relationships in Test 5
4 Validation Parameters and Steps 6
4.1 Example SOAP Envelope with UserName Token 6
4.2 Example Signed SOAP Envelope 7
4.3 Validation Parameters 10
4.4 Validation Steps 10
5 Appendix A: CA List 11

1 Introduction

This document describes Smart Meter Texas (SMT) interface security requirements.

1.1 Prerequisite for Interfaces

This section lists interface security pre-requisites:

- All communication to SMT from Third-Party Service Providers (3rd Party), Retail Electric Providers (REPs) and Transmission Distribution Service Providers (TDSPs) will be over 2-Way Secure Sockets Layer (SSL). For each TDSP, a VPN channel has to be established between SMT and TDSP systems. In the case of TDSPs, SSL will be employed over a virtual private network (VPN).
- For API requests from TDSPs, SMT requires that a user credential be passed in a SAML token that is part of SOAP header. The specific of SAML token is described in a later section of this document.
- For API requests from 3rd Party, REPs, SMT requires that a user credential be passed in a UserName Token that is part of SOAP header. SMT does not require password to be passed as part of UsernameToken. The specifics of UsernameToken are described in a later section of this document.
- SMT will only accept CA issued certificates (SSL and Signer Cert) in its production environment. Self-signed certificates (SSL or Code Signed) will not be accepted.
- It is assumed that partners will use Class 3 certificates.
- The system account for 3rd Party, REPs and TDSPs should exist in the SMT user repository and be known to 3rd Party, REP or TDSP in advance. The system account is case sensitive.
- SMT will only accept signed requests as per WS-Sec 1.1 specification for all the incoming requests. The SMT signature processing module does explicit checks for the signature of UserNameToken, Simple Object Access Protocol (SOAP) Body and Time stamp. Although the SOAP request may be signed, the request will be reject if these elements are not signed.
- Signature Confirmation is not implemented.

2 SMT WS-Security Implementation features

This section describes the implementation of WS-Security by SMT.

- SMT is going to have a Certificate Management System in place. Hence it will use certificates in its certificate store for signature verification purposes. If a certificate is passed as part of the SOAP envelope, the SML Signature processing module ignores it.
- SMT does certificate chain validation before using certificates for Signature validation.
- A SOAP Fault is thrown with an appropriate Fault Code in the event of error.
- The rule processing stops at the action which results into error.
- SMT receives entity information from HTTP header variable named ENTITY_NAME. The value of the header variable should be the Company name of the 3rd Party, REP or TDSP, or the string that uniquely identifies the Company.

3 Trust relationship Establishment

Entities are required to establish a trust relationship with 3rd Party, REPs and TDSPs. Entity creation is an out-of-band process.

3.1 Trust relationships in Production

These entities are required by SMT:

- Signer Certificate – There should be only one CA-issued Signer Certificate that should be given to SMT.
- Value of HTTP header variable ENTITY_NAME.
- SSL Intermediate CA certificate.

This entity is provided by SMT to the 3rd Party or the REP:

- SystemAccount – There will be one SystemAccount that will be created for each 3rd Party and REP. The System Account is used to validate the user against the SMT account database. The System Account for corresponding to the 3rd Party or REP will be provided.

3.2 Trust relationships in Test

These entities are required by SMT:

- Signer Certificate - Self Signed Certificate or CA certificate. The number of this certificate is 1.
- Value of HTTP header variable ENTITY_NAME.
- SSL Certificate – SSL Certificate should be provided to SMT when a 3rd Party or REP is using a self-signed certificate.
- Intermediate CA Certificate – This may be required if a CA certificate is going to be used.

Entity provided by SMT to 3rd Party or the REP:

- SystemAccount – There will be one SystemAccount that will be created for each 3rd Party or REP. The System Account is used to validate the user against the SMT account database. The System Account corresponding to the 3rd Party or REP will be provided

4 Validation Parameters and Steps

4.1 Example SOAP Envelope with UserName Token

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:smt="http://schemas.esb.ams.com/smtxpmessaging">
  <soapenv:Header>
                <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-secext-1.0.xsd">
                 <wsse:UsernameToken>
                          <wsse:Username>REP_ACCT</wsse:Username>
                  </wsse:UsernameToken>
           </wsse:Security>
  </soapenv:Header>
  <soapenv:Body>
     <smt:processPricingMessage>
       <SMTxPPriceSignalRequest>
        <!--Optional:-->
        <RequestID></RequestID>
        <RequesterType>0</RequesterType>
        <RequesterAuthenticationID>957877905</RequesterAuthenticationID>
        <RequesterID>REPAdmin1</RequesterID>
        <RequestPriority>L</RequestPriority>
        <!--Optional:-->
        <CallbackUri></CallbackUri>
        <AddressBlock>
          <!--Optional:-->
          <GroupID></GroupID>
          <!--Optional:-->
          <AddressList>
             <!--0 to 10000 repetitions:-->
             <Address>
              <ESIID>10089010238099798703890</ESIID>
              <MeterSerialNumber>60333057</MeterSerialNumber>[1]
               <!--Optional:-->
              <DeviceMACAddr>PRC-MACADDR1</DeviceMACAddr>
             </Address>
          </AddressList>
        </AddressBlock>
        <PriceMessageBlock>
          <ProviderID>604792792</ProviderID>
          <RateLabel>Rate Label1</RateLabel>
          <IssuerEventID>7001</IssuerEventID>
          <!--Optional:-->
```

[1] <MeterSerialNumber> should be 0 if you are a Third-Party

```
      <CurrentTime>2009-12-14T16:30:00</CurrentTime>
      <UOM>1</UOM>
      <Currency>USD</Currency><PriceTier>1</PriceTier>
      <PriceTrailingDigit>3</PriceTrailingDigit>
      <!--Optional:-->
      <RegisterTier>1</RegisterTier>
      <StartTime>2009-12-18T16:30:00</StartTime>
      <Duration>55</Duration>
      <Price>12777</Price>
      <!--Optional:-->
      <PriceRatio>105</PriceRatio>
      <GenerationPrice>12111</GenerationPrice>
      <!--Optional:-->
      <GenerationRatio>95</GenerationRatio>
      <!--Optional:-->
      <AlternateCostDelivered>1111</AlternateCostDelivered>
      <!--Optional:-->
      <AlternateCostUnit>1</AlternateCostUnit>
      <!--Optional:-->
      <AlternateCostTrailingDigit>2</AlternateCostTrailingDigit>
     </PriceMessageBlock>
    </SMTxPPriceSignalRequest>
   </smt:processPricingMessage>
 </soapenv:Body>
</soapenv:Envelope>
```

- SMT accepts a UserNameToken from 3rd Party or REP with the UserName element. A password is not required.
- The Username element within UserNameToken should be the System Account assigned to the calling 3rd Party or REP. The System Account is set up after the 3rd Party or REPs first admin is registered.
- The RequesterType element should be 0 for REP, or 3 for 3rd Party
- The RequesterAuthenticationId should be the DUNS number for the 3rd Party or REP.

4.2 Example Signed SOAP Envelope

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:smt="http://schemas.esb.ams.com/smtxpmessaging"
 >
  <soapenv:Header>
    <wsse:Security soapenv:mustUnderstand="1"
     xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd"
     >
    <wsu:Timestamp wsu:Id=”Timestamp-6f087de8-475a-4a73-a156-7550dfdb227a”
    xmlns:wsu=”http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
```

```
       1.0.xsd”>
           <wsu:Created>2010-01-19T21:33:16Z</wsu:Created>
           <wsu:Expires>2010-01-19T21:38:16Z</wsu:Expires>
        </wsu:Timestamp>
         <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"
         >
           <SignedInfo>
             <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"
   />
             <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
            <Reference URI="#Id-a73451ca-350b-4ab0-8fb8-4c8a845676fe">
               <Transforms>
                  <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
               </Transforms>
               <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
               <DigestValue>IIADgpAo5apSEJbNFeq84eyK4p4=</DigestValue>
             </Reference>
             <Reference URI="#Id-7bae2f18-a07b-4602-bc3d-237765fd72d0">
               <Transforms>
                 + <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
               </Transforms>
               <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
               <DigestValue>Dw97YWmyoFZ1zEHdfu6yR9dY7Zs=</DigestValue>
             </Reference>
             <Reference URI="#Timestamp-6f087de8-475a-4a73-a156-7550dfdb227a">
               <Transforms>
                  <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
               </Transforms>
               <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
               <DigestValue>ol1HC0SNO3lZldrEaOSxGUFzFn4=</DigestValue>
             </Reference>
           </SignedInfo>
           <SignatureValue>gJbQeqjQYVwcLjTRj30y1u8xEhnClbPeNUL1Ky83rl6f+ZVjmuBBnAyJ3SHLsD
           xi/DXfcKCbsPagSGAxcxch0/HPP+yT6bdIMiLWd/lkGSipBETfWWkeWfAKBLoIQCorbE+j2FxNP
           KVR/mU1nrg2iVV9c  LM8Bwy2Mu90U969r3c=</SignatureValue>
           <KeyInfo>
             <wsse:SecurityTokenReference>
               <dsig:X509Data xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
               >
                 <dsig:X509IssuerSerial>
                   <dsig:X509IssuerName>CN=www.rxyz.com, OU=Retail, O=RXYZ, L=Dallas,
   ST=TX, C=US</dsig:X509IssuerName>
                   <dsig:X509SerialNumber>1262968678</dsig:X509SerialNumber>
                 </dsig:X509IssuerSerial>
               </dsig:X509Data>
             </wsse:SecurityTokenReference>
        </KeyInfo>
```

```
                </Signature>
<wsse:UsernameToken wsu:Id="Id-a73451ca-350b-4ab0-8fb8-4c8a845676fe" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
   <wsse:Username>CNP</wsse:Username>
</wsse:UsernameToken>
              </wsse:Security>
          </soapenv:Header>
         <soapenv:Body wsu:Id="Id-7bae2f18-a07b-4602-bc3d-237765fd72d0"
          xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
     1.0.xsd">
              <smt:processSimpleMessaging>
                  <SMTxPSimpleMessageRequest>
                    <!-- Optional: -->
                    <RequestID />
                     <RequesterType>0</RequesterType>
                     <RequesterAuthenticationID>799530220</RequesterAuthenticationID>
                     <RequesterID>STAGEREPFIRSTADMIN16JAN</RequesterID>
                     <RequestPriority>M</RequestPriority>
                     <!-- Optional: -->
                     <CallbackUri />
                     <AddressBlock>
                        <GroupID/>
                        <AddressList>
                          <Address>
                            <ESIID>10089010091306094601OO</ESIID>
                            <MeterSerialNumber>0</MeterSerialNumber>
                            <DeviceMACAddr>001DB7000001D7F9</DeviceMACAddr>
                          </Address>
                          </AddressList>
                     </AddressBlock>
                     <SimpleMessageBlock>
                         <MessageID>40004</MessageID>
                         <StartTime>2010-01-19T16:39:00</StartTime>
                         <DurationTime>5</DurationTime>
                         <Message>Simple Message TEST 2010-01-19 (Central Time)</Message>
                         <MCTransmission>0</MCTransmission>
                         <MCPriority>0</MCPriority>
                         <MCConfirmation>0</MCConfirmation>
                     </SimpleMessageBlock>
                 </SMTxPSimpleMessageRequest>
             </smt:processSimpleMessaging>
         </soapenv:Body>
     </soapenv:Envelope>
```

- SMT accepts signed message from its partner. In the event that it receives an unsigned message or message with invalid signature, the request will be rejected and SOAP Fault will be sent.
- For signed messages, SMT mandates that the SOAP Body, UsernameToken and TimeStamp elements

be explicitly signed as in the example above. If anyone of the elements is not signed, the request will be rejected and SOAP Fault will be sent.

- The Token reference type is IssuerSerial. SMT will not extract a certificate from the SOAP Header for Signature validation. It uses certificates from its certificate store for Signature Validation. Therefore, it is imperative that certificates are exchanged prior to commencement of transaction.

4.3 Validation Parameters

Following table describes validation parameters, source and purpose:

Parameter	Source	Purpose
ENTITY_NAME	HTTP header	To know the originator of request
UserName	Child element of UserNameToken element of SOAP Header	To validate against System Account
RequesterType	Child element of SOAP Body	To get a LDAP branch where System Account will be validated.
RequesterAuthenticationID	Child element of SOAP Body	To validate the DUNS number that is presented in this element.
Signer Certificate	Out-of-Band	To validate the signature
Intermediate SSL Certificate	Out-of-Band	SSL handshake

4.4 Validation Steps

Following are the authentication and validation sequence that happens on the SMT perimeter:

- SMT performs 2-way SSL hand-shake with the 3RD Party or REP's endpoint.
- SMT gets the 3rd Party or REP's name or string identifying the 3rd Party or REP by reading the HTTP Header variable.
- SMT performs Schema Validation.
- SMT validates the signature of the incoming request.
- SMT validates the system account of the 3rd Party or the REP using the HTTP header variable ENTITY_NAME, UserName from UserNameToken, and RequesterType elements.
- SMT validates the DUNS number of the 3rd Party or the REP using the HTTP header variable ENTITY_NAME, RequesterType and RequsterAuthenticationID elements

If all the steps are successful, the request is sent to SMT internal systems. In the event of failure at any step, the fault message is sent with an appropriate fault code.

5 Appendix A: CA List

SMT will accept certificates (SSL and Signer) issued by following Certificate Authorities:

American Express

ANX

Belacom-E-Trust

C-and-W-HKT-SecureNet-CA

Certipose

Certisign

Certplus

Deutshe

Entrust

Equifax

EUnet

FESTE

First-Data

GlobalSign

GTE-CyberTrust

Microsoft

NetLock

RSA

Saunalahaden

SecureNet

SecureSign

SwissKey

TC-TrustCenter

Thawte

UTN-DATACORP

Valicert

Verisign

ViaCode