CICS Web Service Security

Anthony Papageorgiou
IBM
CICS Development
March 13, 2012
Session: 10282

Agenda

- CICS Web Service Support Overview
- Security Basics and Terminology
- Pipeline Security Overview
- Identity
- Encryption
- Signature
- DataPower
- Identity Propagation

Security Basics and Terminology

- CICS Security is based on User ID
  - Command Level Security
  - Resource Level Security
  - etc.
- The various options for securing Web Services in CICS are aimed at addressing three common needs:
  - Identity
  - Confidentiality
  - Integrity

Identity

- As a Web Service Provider:
  - Need to know who the requester is so that we know what User ID we should run the business logic under so that CICS Security is maintained
  - Authentication – Need to prove that the requester is who they say they are.
- As a Web Service Requester:
  - Need to know what credentials we should provide to the Web Service Provider
Confidentiality

- Web Services allow data in COMMAREA or Channels to travel outside of CICS via SOAP Messages
- Need to ensure that the data in our SOAP messages can only be read by the intended recipient

Integrity

- Web Services allow instructions to business logic to travel outside of CICS via SOAP messages
- Need to ensure that the data in our SOAP messages cannot be tampered with without us knowing

Pipeline Security Overview

CICS as a Web Service Provider

CICS as a Web Service Requester

Identity

Identity Overview

"Where we get the User ID from..."
CICS as a Web Service Provider
Identity Overview

“What credentials to send...”

CICS as a Web Service Requester

Identity Scenarios

- Identity assigned on a per service basis
- Identity assigned on a per requester basis (basic)
- Identity assigned on a per requester basis (advanced)
- Identity from external credentials
- Identity from X.509 certificates

Identity Per Service

CICS as a Web Service Requester

For CICS TS V4.1 and above, the XWBAUTH user exit is called to determine username and password for basic authentication credentials given the current CICS User ID and the URI that is being accessed.

Security handler determines identity credentials from pipeline configuration and adds them to the header of the SOAP message. This may or may not be dependent on the current CICS User ID.

Can also use the CERTIFICATE(...) attribute if using SSL.

Identity Per Requester (basic)

CICS as a Web Service Requester

- As of CICS TS V3.2 for HTTP a check will be made for a Client Mode URIMAP when making an outbound connection. The properties of this URIMAP will be used including Certificate and Ciphers if SSL.
- Can be achieved with a user handler that updates DFHWS-USERID.

Identity Per Service

CICS as a Web Service Requester

- HTTP Basic Auth
- SSL Client Certificate Authentication
- Both configured on TCPIPSERVICE

Identity Per Requester (basic)

CICS as a Web Service Requester

- As of CICS TS V4.1 the XWBAUTH exit is called
- Can be achieved with a user handler

URIMAP can be used to assign a User ID for the pipeline task. If the User ID (or indeed Tran ID) can be derived directly from the URI of the Service this is the most efficient option.

Custom handler could be used, per pipeline, but if this makes a static per pipeline decision then URIMAP is more efficient.
A typical pattern is to have a trusted gateway that sends Web Service requests into CICS on behalf of various users. This means that users of a service only need security permissions to run the business logic (eg file/program access) but not to connect to CICS.

For this scenario you need CICS Web service security support.

Transport level security is used to identify the trusted server.

So we cannot use this to identify the actual requester.

The Trusted Portal puts the credentials of the requester into the SOAP request header.

The Web Service Security handler extracts these credentials and assigns the appropriate CICS User ID.

NOTE: The Trusted Portal’s ID must have surrogate authority for the requester’s User ID.

CICS supplied security handler
- Simple case: CICS User ID in SOAP header
- With password/username. Mode = Basic - Authenticate actual requester in CICS
- Without password: Mode = Basic, Trust = Blind - Authenticate actual requester in trusted portal
- Custom Handler can be used

Makes sense in CICS TS 3.1

Example Pipeline Configuration

CICS User ID in SOAP Header

```xml
<wsse_handler>
    <dfhwssconf version="1">
        <authentication mode="basic" trust="blind">
            <username></username>
            <password></password>
        </authentication>
    </dfhwssconf>
</wsse_handler>
```
Identity from External Credentials

- External Credentials are ones that cannot be handled natively by CICS
  - External user id / passwords
  - SAML, LPTA, Kerberos
  - Home grown etc.
- CICS as Requester and Provider
  - WS-Trust support to call and external Secure Token Service
  - Custom Handler

WS-Trust Specification

- WS-Trust
  - Published as specification 25 February 2005
  - Submitted to OASIS standardization process
  - Provides a framework for building trust relationships
    - Sender and Receiver in different security domains
    - Security tokens must be vouched for by trusted third party
    - Trusted third party is called the Security Token Service (STS)
  - WS-Trust defines standard protocols and standard WSDL interfaces to communicate with an STS

CICS Support of WS-Trust Options

- Interoperates with a Security Token Server
  - CICS supplied security handler
    - CICS as a Web Service provider
      - Validate the security token in the WS-Security header
      - Exchange the security token in the WS-Security header
    - CICS as a Web Service requester
      - Exchange the security token to be used in the WS-Security header
  - Custom interaction with an STS
    - CICS Provides a Channel Linkable interface to allow user programs to easy call an STS, without understanding WS-Trust
    - Via DFHIPRT
    - CICS Builds and parsers the WS-Trust messages to and from containers

Example Pipeline Configuration

```xml
<wse_handler>
  <dfhwse_configuration version="1"
    <sts_authentication action="Issue">
      <auth_token_type>
        <namespace>http://...</namespace>
        <element>MyToken</element>
      </auth_token_type>
    </sts_authentication>
    <sts_endpoint>
      <endpoint>https://...</endpoint>
    </sts_endpoint>
  </dfhwse_configuration>
</wse_handler>
```
Identity from X.509 Certificates

- Datapower
  - Transform to simple identity token in Data Power and use mode = basic, trust-blind
  - Optionally with SSL
- CICS Supplied security handler
  - Will use the identity associated with the X.509 certificate used to sign the message body (i.e. Via RACF Keyring)
  - Very CPU heavy, so should only be used as a last resort or for low volume transactions

Example Pipeline Configuration

```xml
<wsse_handler>
  <dfhwsse_configuration version="1">
    <authentication mode="signature">
      <algorithm>
        http://www.w3.org/2000/09/xmldsig#dsa-sha1
      </algorithm>
      <certificate_label>
        MY_CERTIFICATE_LABEL
      </certificate_label>
    </authentication>
  </dfhwsse_configuration>
</wsse_handler>
```

Identity from X.509 Certificates

- More Advanced Identity Scenarios
  - Multiple Identity Tokens
    - Asserted Identity
      - CICS can natively handle X.509 and user name tokens
      - Trust = basic/blind/signature can be used to specify asserting (checked) ID.
      - Mode = basic/signature can be used to specify asserted (unchecked) ID.
      - Surrogate Security checks are used to ensure that the Asserting ID has authority to start work for target asserted user ID.

Encryption

- Get identity from the SSL digital signature
- For outbound: The hashing algorithm to use to sign
- For outbound: The certificate to use to sign

More Advanced Identity Scenarios

- Multiple Identity Tokens
  - Asserted Identity
    - CICS can natively handle X.509 and user name tokens
    - Trust = basic/blind/signature can be used to specify asserting (checked) ID.
    - Mode = basic/signature can be used to specify asserted (unchecked) ID.
    - Surrogate Security checks are used to ensure that the Asserting ID has authority to start work for target asserted user ID.
Encryption

- Data must be encrypted between Requester and Provider, provides Confidentiality
  - If calls are point to point then use transport encryption (SSL)
  - If service requires WS-Security XML Encryption then you can use DataPower
  - If WS-Security XML Encryption is required and DataPower is not available then the CICS supplied security handler can be used
    - Can decrypt any elements in an inbound message
    - Can only encrypt the body on an outbound message

Example Pipeline Configuration

```
<wsse_handler>
  <dfhwsse_configuration version="1">
    <encrypt_body/>
    <http://www.w3.org/2001/04/xmlenc#tripledes-cbc/>
    <algorithm>http://www.w3.org/2001/04/xmlenc#tripledes-cbc</algorithm>
    <certificate_label>ENCCERT02</certificate_label>
    <expect_encrypted_body/>
  </dfhwsse_configuration>
</wsse_handler>
```
Sample XML

```xml
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>4019 2445 0277 5567</Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```

Signature Signing with Public/Private key pair

A certificate containing the public key is included in the message header.

Signature “How we stop our data changing...”

For Outbound:
- The algorithm with which to sign the message body
- The certificate with which to sign the message body
- Flag to reject messages which don’t have a signed body

Example Pipeline Configuration

```xml
<wsse_handler>
  <dfhwsse_configuration version="1">`
    `<sign_body>`
      `<algorithm>`
          `http://www.w3.org/2000/09/xmldsig#rsa-sha1`
      `</algorithm>`
      `<certificate_label>`SIGCERT01`</certificate_label>`
  `</sign_body>`
  `<expect_signed_body/>`
</dfhwsse_configuration>
```

Signature

- Data must not have changed since it was sent, Integrity.
  - If encryption is used then a weak form on Integrity is implied
    - It is hard to make meaningful changes to encrypted data
    - In such cases again SSL may be enough
  - If service requires XML Digital Signature then you can use DataPower
  - If DataPower is not available you can use the CICS supplied Security Handler
    - Can verify signature on any elements in an inbound message
    - Can only sign the body on an outbound message
**Identity Propagation**

- New in CICS TS V4.1
  - z/OS 1r11
  - PK5579, PK8741 & PK98426
- Enables two way mapping between dname@realm userid and RACF user ID
- Allows task association data to include BOTH RACF user and ICRX data
- Map an identity through the entire enterprise

**DataPower**

*Some more use cases...*

Monitoring and control
Example: centralized ingress management for all Web Services using ITCAM SDA
Deep-content routing and data aggregation
Example: XPath (content) routing on Web Service parameters
Functional acceleration
Example: XSLT, WS Security
Application layer security and threat protection
Example: XML Denial-of-Service protection, WS Security

**Summary**

- CICS Web Service Security Support Overview
  - Identity – Transport and Message Level
    - Native (CICS User ID) Security tokens
    - WS – Trust for non native security tokens
    - X.509 Certificates
  - Encryption – Transport and Message Level
    - SSL for Transport level
    - Inbound: XML element encryption for message level
    - Outbound: Whole body encryption for outbound (XML element with data power)
  - Signature – Message Level only
    - Inbound: XML element level signing
    - Outbound: Whole body signing for outbound (XML element level with data power)
  - Recommend using Data Power
- Identity Propagation
  - Propagate originating identity through CICS with ICRX

---

**DataPower Integration Appliance Xi50**

- DataGlue “Any-to-Any” Transformation Engine
- Content-based Message Routing: Message Enrichment
- Protocol Bridging (HTTP, MQ, JMS, FTP, IMS Connect, etc.): Request-response and sync/async matching
- Direct to Database: Communicate directly with remote Database instances
- XML/SOAP Firewall: Filter on any content, metadata or network variables
- Data Validation: Approve incoming/outgoing XML and SOAP at wire speed
- Field Level Security: WS-Security, encrypt & sign individual fields, non-repudiation
- XML Web Services Access Control/AAA: AML, LDAP, RADIUS, etc.
- MultiStep: Sophisticated multi-stage pipeline
- Web Services Management: Centralized Service Level Management, Service Virtualization, Policy Management
- Easy Configuration & Management: Net GUI, CLI, IDE and Eclipse configuration to address broad organizational needs
Google us or check us out at:

- ibm/developerworks/cicsdev
- facebook.com/IBM_CICS
- twitter.com/IBM_CICS
- youtube.com/cicsfluff
- youtube.com/cicsexplorer
- themasterterminal.com
- twitter.com/IBM_System_z
- IBM CICS Explorer Forum ibm.com/developerworks/forums/forum.jspa?forumID=1475&start=0
- CICS-L list Forum listserv.uga.edu/archives/cics-l.html

Thank you,
Any Questions?

http://atlanta.SHARE.org/SessionEvaluation
Session: 10282