Building a Protocol validator for B2B Communications

Rudi van Drunen
Rix Groenboom
Protocol validator | Agenda

• Business problem
• Solution
• Architecture
• Building (certificates)
• Issues encountered
• Lessons learned
Validator | The business problem

• Deregulation of the Dutch Utility market

• Communication structure transition
  – Peer to Central
  – Peer-to-Peer

• How do we make sure this is working?
Solution | Certification

• Make sure all comms work
  – Certify all communication paths

• One central place for certification
  – All peers need to obtain certification
  – Protocol run-through
  – Both directions
• Define the protocol
  – Define the test set (roles)

• Establish a PKI

• Build the infrastructure and software

• Have all partners go through the procedure
  – All peers need to obtain certification
Validator | The Workflow

Diagram showing the workflow process with nodes for xml Message, Send Message, Receive Message, Send Message, Receive Message, Send Message, Control, and Validation.
• Message standard
  – Messages are encoded in XML
  – Transported over as2 (MIME encoded)
    • Authentication and Encryption
  – Wire protocol: https
System | Components

- Frontend
  - Ultra ESB (Adroitlogic)
  - Open Source Enterprise Service Bus

- Validator
  - Virtualize (Parasoft)
  - Service virtualization component

- Protocol analyzer: custom Python

- GUI: custom Java
System | Infrastructure

- Windows 2008 server
- MySQL
- Eclipse
- Python
- Firewall tools
Validator | Tooling

- Eclipse development environment
- Mendelsson as2 simulator
- Keystore Explorer
• sync / async messages
  – extra http header (gnd)
• Sequence of messages
  – validator code / database
• PKI
  – certificate aliases
  – multiple levels / keystores
Building | Certificates (1)

• Wire level
  – https server certificate
    • data (as2 endpoint) channel
    • control (web portal) channel

• Global PKI and CA
  – Authentication of server
  – Encryption of data
• as2 level

• Private PKI
  – Authentication of trading partner
  – Encryption of the payload
    • MIME encoding
  – Make sure the partner has the right CA
• Test using Mendelsson
  – as2 simulator
  – Technical connection test
    • connect to frontend bus
    • send an (encrypted) message
    • receive an (encrypted) message

• Management of keystores
  – different keystores, different certificates, aliases
Building | Experiences

• Rapid prototyping
  – Off-the-shelf components
  – Not much coding needed

• Protocol was not completely fixed
  – Definition of test messages (xml)
  – Definition of test scenarios (sequence)
• Certificates
  – 2 levels
    • 2-step debugging
• PKI is difficult to understand
  – Certificate aliases
Building Experiences

• Fast turn-around time
  – 3 weeks from start to prototype

• Communication to a lot of partners
  – Different backends

• Bugs in the components
  – Java logging / messages
Building | Lessons learned

• Architecture is important
  – modular design
  – reuse in mind
• COTS components
  – selection of the components
• XSD changes
• PKI
Building | Lessons learned

• Infrastructure
  – readyness, completeness

• Time schedule
  – Allow for debugging while in operation

• External partners
  – All wait until the last moment to get certified
  – All have different systems
  – Communication
Validating B2B communications in a deregulated utility business

- Rapid prototyping using COTS
- Flexible in Protocol
  - re-usable
  - **Generic** system for protocol validation

As of 11/20/2012 all partners (>36) certified.
Questions | Lessons learned

R.van.Drunen@xs4all.nl