Message Bus Refactor Using RabbitMQ

September 25, 2019

Shwetha Niddodi
Software Engineer
Agenda

• Need for message bus upgrade
• Difference between ZeroMQ VOLTTRON and RabbitMQ VOLTTRON
• RabbitMQ VOLTTRON overview
• Connecting to remote platform using CSR
• Deployment use case example
• Integration with third party tools
Message Bus Upgrade

• VOLTTRON’s ZeroMQ based message bus has been key for meeting the security and interoperability goals of the platform

• At the same time, RabbitMQ has become more mature as it has seen major investment by commercial companies.
  ▪ Used by: Instagram, Indeed.com, Google Cloud Platform, Tesla …

• Goals of the Refactor:
  ▪ Maintain essential features of current message bus and minimize transition cost
  ▪ Leverage an existing and growing community dedicated to the further development of RabbitMQ
  ▪ Move services provided currently by VOLTTRON agents to services natively provided by RabbitMQ
  ▪ Decrease VOLTTRON development time spent on supporting message bus which is now a commodity technology.
  ▪ Address concerns from community about ZeroMQ

• View this effort as essential to the long-term future of the platform
  ▪ Working with heavy users in the community to get feedback
  ▪ Reduce long term costs of platform by moving message bus development out of core
  ▪ Maintain support for ZMQ short term (3 – 5 years) as funding allows
### ZeroMQ Based VOLTTRON vs RabbitMQ Based VOLTTRON

<table>
<thead>
<tr>
<th>Feature</th>
<th>ZeroMQ Based VOLTTRON</th>
<th>RabbitMQ Based VOLTTRON</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform acts as the broker and is responsible for routing the messages</td>
<td>Separate broker running outside the platform and all agents connect to that broker. Exchanges are responsible for routing.</td>
<td></td>
</tr>
<tr>
<td>Authentication is based on ZAP protocol using Elliptical Cryptographic Curve Key</td>
<td>SSL based authentication using TLS X509 certificates</td>
<td></td>
</tr>
<tr>
<td>Remote agent authentication is by adding public key of remote agent to auth.json</td>
<td>Remote agent authentication is through Certificate Signing Request operation</td>
<td></td>
</tr>
<tr>
<td>Custom agents such as forward historian agent for forwarding messages from one platform to another</td>
<td>Shovel plugin can be used for same purpose</td>
<td></td>
</tr>
<tr>
<td>VOLTTRON specific implementation for multi-platform connection</td>
<td>Federation plugin can be used for same purpose</td>
<td></td>
</tr>
<tr>
<td>Custom agents to connect to third party tools such as MQTT historian agents</td>
<td>Easy integration with third party tools - MQTT - ElasticSearch</td>
<td></td>
</tr>
<tr>
<td>Need to build custom agent that monitors status of message bus</td>
<td>Monitor message bus status such as message rates, resource usage of queue and data rates of client connections</td>
<td></td>
</tr>
<tr>
<td>Scalable multi-platform connections</td>
<td>Highly scalable – does not require $O(n^2)$ connections between n brokers</td>
<td></td>
</tr>
<tr>
<td>Less flexibility in deployment compared to RabbitMQ</td>
<td>Flexibility in deployment</td>
<td></td>
</tr>
</tbody>
</table>
RabbitMQ Overview

- RabbitMQ uses AMQP (Advanced Message Queuing Protocol)
- Exchanges – Responsible for routing of messages to Queues.
- Queues - Buffer that stores the messages until consumed by consumer.
- Bindings – Queues bind to the exchange with binding keys
  - Messages are routed based on bindings
Pubsub in RabbitMQ VOLTTRON

Agent_A
VIP ID: “agent_a”

VIP queue
Binding Key: “volttron1.agent_a”

Publish Message:
Routing Key:
“__pubsub__volttron1.devices.hvac1”
Pika properties:
Type:“pubsub”
user_id: “volttron1.agent_a”
Message_id: result.ident
Body: message arguments

Topic Exchange

Agent_B
VIP ID: “agent_b”

Subscribe:
Pubsub queue: “volttron1.*”
Routing Key:
“__pubsub__volttron1.devices.#”
Message_id: message id
Callback handler: user pubsub message handler

User PUBSUB message handler
Security Feature With RabbitMQ VOLTTRON

- RabbitMQ supports multiple authentication mechanisms
  - For VOLTTRON we use SSL peer verification using with x509 certificates
- SSL certificates of interest
  - Root CA
  - Server certificate signed by Root CA
  - Client certificate signed by Root CA
RabbitMQ-VOLTTRON Authentication Using SSL

- **Agent 1**
  - `agent1.crt` – public cert signed by Root CA
  - `agent1.pem` – private key

- **Agent 2**
  - `agent2.crt` – signed by Root CA
  - `agent2.pem` – private key

- **Agent 3**
  - `agent3.crt` – signed by Root CA
  - `agent3.pem` – private key

**RabbitMQ Server**
1. Root CA File - CA1 public key
2. Server cert signed by Root CA
3. Server key

Presents `agent1.crt` and `agent1.pem`
Presents `agent2.crt` and `agent2.pem`
Presents `agent3.crt` and `agent3.pem`
Multi-Platform Connection With SSL Certificates

Multi-Platform Connection with SSL certificates

VOLTTRON 1

Rabbitmq Server - 1
1. CA cert file – contains CA1 public key and CA2 public key
2. Server cert signed by CA1
3. Server private key

Presents agent1.crt and agent1.pem

Agent 1
agent1.crt – signed by Root CA1
agent1.pem – private key

VOLTTRON 2

Rabbitmq Server - 2
1. CA cert file – contains public key of CA2 public key and CA1 public key
2. Server cert signed by CA2
3. Server private key

Presents agent2.crt and agent2.pem

Agent 2
agent2.crt – signed by Root CA2
agent2.pem – private key

Shovel

Presents voltron2 user cert, key

Federation

Presents voltron2 user cert, key
Connecting To Remote Platform Using CSR

**Rabbitmq Server - 1**
1. CA cert file - contains public key of CA1
2. Server cert signed by CA1
3. Server private key

Presents agent1.crt and Agent1.pem

**VOLTTRON 1**

**Remote Agent**

**Rabbitmq Server - 2**
1. CA cert file – contains public key of CA2
2. Server cert signed by CA2
3. Server private key

**VOLTTRON 2**

Discovery Info Request

- CA cert, instance name, RMQ address
- CSR Request
- Signed public cert

/discovery

/csr/request_new
CSR Admin

• CSR Admin Login – URL: https://<hostname>:8443

![CSR Admin Login](image)

• CSR Request

![CSR Request](image)
CSR Admin

- CSR Approved

Certificate Requests

APPROVED for central.central.platform.agent

- Deny Delete
- Status: APPROVED
- Common Name: central.central.platform.agent
- Remote IP: 172.20.214.72

- CSR Denied

Certificate Requests

DENIED for central.collector.platform.agent

- Deny Delete
- Status: APPROVED
- Common Name: central.central.platform.agent
- Remote IP: 172.20.214.72

- Approve Delete
- Status: DENIED
- Common Name: central.collector.platform.agent
- Remote IP: 172.20.214.65
Multi-Platform Multi-Bus Connection

RMQ Central

- CSR request
- CSR response

RMQ Collector

Auth entry in auth.json

ZMQ Collector

Forwarder, VCP, Master driver

VC, Historian
Volttron-ctl Commands For Creating Certificates

• Create a volttron public cert and private key for a new client
  volttron-ctl certs create-ssl-keypair jackpot
  
  New public cert will be $VOLTTRON_HOME/certificates/certs/<instance-name>.jackpot.crt
  New private key will be $VOLTTRON_HOME/certificates/private/<instance-name>.jackpot.pem

• Export the private key / cert into a PKCS12 format file
  volttron-ctl certs export-pkcs-12 jackpot jackpot.12
New Options For VOLTTRON Users

• Offers more flexibility in deployment
  ▪ Federation
  ▪ Shovel
  ▪ Highly scalable
    ✓ Connect numerous buildings spread over large geographical area.
Deployment Use Case Example

Campus\textsubscript{1}

Building\textsubscript{1}

Building\textsubscript{2} 

Building\textsubscript{3}

ML app

2-way Federation link

Shovel

Occupancy 
Comfort

Energy 
Cost

Building level status/control action

Zone level status/control action

Device Status and control action

HVAC 
Lighting 
Smart plug

Control Action

Building status
Integration with 3\textsuperscript{rd} party tools/applications

- Integration with non VOLTTRON RabbitMQ clients
  - Client connected to same broker
  - Connected to different broker. Connection established using federation/shovel
- ElasticSearch – For data ingestion and to perform analysis
  - ElasticSearch – For data ingestion and to perform analysis
    - Cybersecurity – to detect anomalies in data etc.
    - Visual Analytics
  - MQTT – For cloud based applications
- Github link: \url{https://github.com/VOLTTRON/external-clients-for-rabbitmq}
Next Steps

- Integrate federation and shovel setup with CSR
- Ease of deployment based on user feedback
Thank you