Design for Security Message Bus (Scratch Document)

# Summary

The Security Message Bus is designed to ingest messages from the various Cloud Portals systems in leverage Service Now Workflow to open tickets and allow for updates and closure of tickets in both platforms.

Data will be handled via RabbitMQ and MariaDB. RabbitMQ will be used to move messages from the various service buses into processing for being stored in MariaDB and into Service Now Workflow.

The infrastructure will be deployed and scaled within Kubernetes.

[Diagram]

## RabbitMQ Message Format

Messaged contain an external routing layer that then contains the raw message from the service it is picking up data from. The external routing leverage ‘:’ for separation of the various elements with the interior or raw message separated with ‘,’.

Queue for this service will be uni directional with message either designated at Inbound ‘I’ or Outbound ‘O’. Inbound is defined as from the security portal to message processor and Outbound is message processor to security portal.

We will keep a similar structure for both message direction as this layer is used for routing of message to or from subscribers and publisher that handled the last mile.

The interior message or message body is based on the cloud service from which we are ingesting from or responding to, and the routing to the processor is determined by the messageType field.

queueName : direction : clientID : clientAcct : messageType : messageBody

* queueName = CloudVendor
* direction = I / O
* clientID = ClientID / ClientIDSub
* clientAcct = Client Account ID
* messageType = SecurityPortal (ex: Sentinel)
* messageBody = Service Message

## Message Processor

For each messageType we will create a focused processor to handle either storage and reformat for outbound.q

[Example Code]

## Required PIP Packages

Azure [https://docs.microsoft.com/en-us/azure/developer/python/sdk/azure-sdk-library-package-index]

AWS Client [https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/eb-cli3-install-windows.html]

RabbitMQ [https://pypi.org/project/rabbitmq/] [https://pypi.org/project/pika/]

Influxdb [https://github.com/influxdata/influxdb-python]

Mongodb [https://www.mongodb.com/languages/python]

Mariadb [https://mariadb.com/resources/blog/how-to-connect-python-programs-to-mariadb/]

AWS Client []

## Testing Environment

This will be deployed and testing on a single AWS Ubuntu Instance that will deploy all components:

1. Python 3.09
2. AWS Client
3. Azure Client
4. MariaDB
5. RabbitMQ

### Service NOW SaaS Development Environment

### Azure Cloud SaaS Security Components

### AWS Cloud Infrastructure Components