

MQ Monitoring on Cloud

Suganya Rane

Digital Automation, Integration & Cloud Solutions

Agenda

- ☐ Metrics & Monitoring
- ☐ Monitoring Options
- ☐ AWS ElasticSearch
- ☐ Kibana
- ☐ MQ CloudWatch on AWS
- ☐ Prometheus
- ☐ Grafana
- ☐ MQ on AWS Options

Metrics & Monitoring

- MQ V9 makes many statistics available through a pub/sub interface
- Option to remotely subscribe to topics under \$SYS/MQ for information on:
 - CPU usage
 - Disk usage
 - Connections and disconnections
 - Opening and closing of queues
 - Pub/sub and put/get
 - Syncpoint calls
 - Changes to MQ objects (MQSET and MQINQ)

Metrics & Monitoring

- Publish to remote metrics servers like Graphite, Prometheus
 - Visualize using Grafana

What can we monitor?

IBM MQ contains seven key elements that are critical to the overall functionality and availability of the application:

- Queue Managers
- Queues
- Cluster Queue Managers
- Channels
- Listeners
- Service Objects
- Events

Monitoring Options on the Cloud

- Use the AWS ElasticSearch service (forward logs).
- You could also forward your logs to a third-party log service, such as Splunk, Loggly, Elastic Cloud, or the IBM Bluemix Logmet service.
- Use Amazon's own service, CloudWatch.

AWS ElasticSearch

The ElasticSearch stack is a useful way to store, search and visualize data.

The ELK stack comprises 3 components:

- ElasticSearch - A distributed search engine
- Logstash - a log collection & processing tool
- Kibana - a browser based visualization tool for displaying and searching log data

AWS ElasticSearch

1. Install your own ElasticSearch
2. Install your Kibana server (The above two have to be installed & ready to send data to them).
3. Configure Logstash to store & process MQ logs (i.e. AMQERR*.LOG)
4. Use Kibana to view & search through those logs.

Sending logs to Elasticsearch

- Logs are written into the Elasticsearch engine by Logstash.
- Consolidate logs from multiple machines producing log records.
- Logstash needs to get the log records from those machines, process them to filter out unwanted data, and send them to Elasticsearch.

Sending logs to Elasticsearch

Getting logs from each of your machines into Elasticsearch is done in one of two ways:

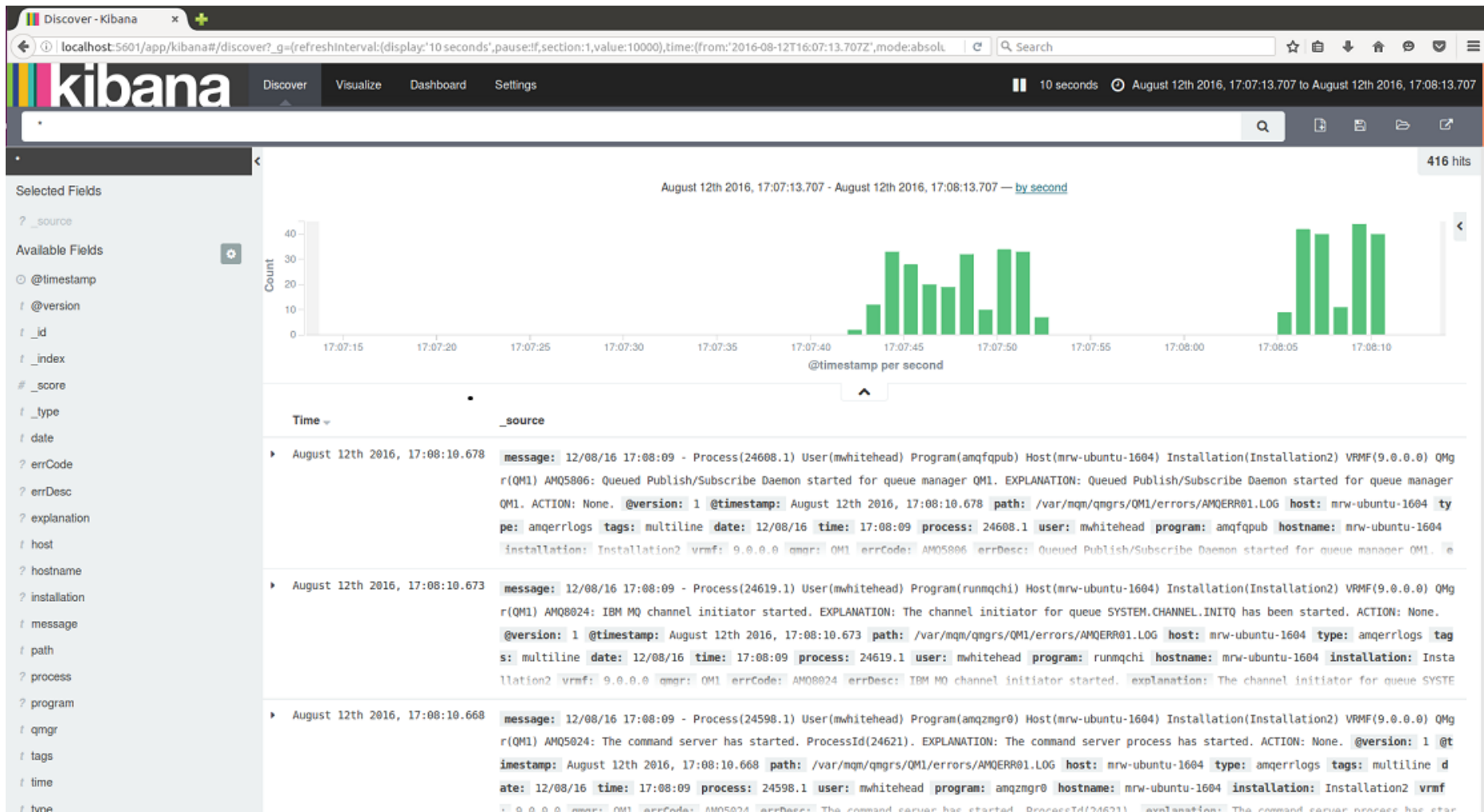
1. Sending the complete log to a central Logstash agent, where it is processed
2. Running a Logstash agent on each machine and processing them locally

****Refer to Matt Whitehead's blog on details of how to do this.**

Kibana Dashboards

After having the MQ records being written to Elasticsearch we can use Kibana to view and search through them.

Kibana Dashboards



Kibana Dashboards

The screenshot shows the Kibana Discover interface. The left sidebar contains a list of fields: `_source`, `_type`, `date`, `errCode`, `errDesc`, `explanation`, `host`, `hostname`, `installation`, `message`, `path`, `process`, `program`, `qmgr`, `tags`, `time`, `type`, `user`, and `vmf`. The main area displays a log entry for August 15th, 2016, at 09:45:53.403. The log message is: "15/08/16 09:45:52 - Process(26913.1) User(mwhitehead) Program(runmqchl) Host(mrw-ubuntu-1604) Installation(Installation2) VRMF(9.0.0.0) QMgr(QM1) AMQ9002: Channel 'QM2' is starting. EXPLANATION: Channel 'QM2' is starting. ACTION: None. @version: 1 @timestamp: August 15th 2016, 09:45:53.403 path: /var/mqm/qmgrs/QM1/errors/AMQERR01.LOG host: mrw-ubuntu-1604 type: amqerrlogs tags: multiline date: 15/08/16 time: 09:45:52 process: 26913.1 user: mwhitehead program: runmqchl hostname: mrw-ubuntu-1604 installation: Installation2 vmrf: 9.0.0.0 qmgr: QM1 errCode: AMQ9002 errDesc: Channel 'QM2' is starting. explanation: Channel 'QM2' is starting. id: AVaNY0q1G7XHmVRAy34 type: %[@metadata][type] index: %[@metadata][type]-2016.08.15/%[@metadata][type]/AVaNY0q1G7XHmVRAy34". Below the log entry, there is a table view showing the fields and their values. The table has columns for field names and values. The fields are: `@timestamp`, `@version`, `_id`, `_index`, `_score`, `_type`, `date`, `errCode`, `errDesc`, `explanation`, `host`, `hostname`, `installation`, `message`, `path`, `process`, `program`, `qmgr`, `tags`, and `time`. The values are: August 15th 2016, 09:45:53.403, 1, AVaNY0q1G7XHmVRAy34, %[@metadata][type]-2016.08.15, %[@metadata][type], 15/08/16, AMQ9002, Channel 'QM2' is starting., Channel 'QM2' is starting., mrw-ubuntu-1604, mrw-ubuntu-1604, Installation2, 15/08/16 09:45:52 - Process(26913.1) User(mwhitehead) Program(runmqchl) Host(mrw-ubuntu-1604) Installation(Installation2) VRMF(9.0.0.0) QMgr(QM1) AMQ9002: Channel 'QM2' is starting. EXPLANATION: Channel 'QM2' is starting. ACTION: None., /var/mqm/qmgrs/QM1/errors/AMQERR01.LOG, 26913.1, runmqchl, QM1, multiline, 09:45:52.

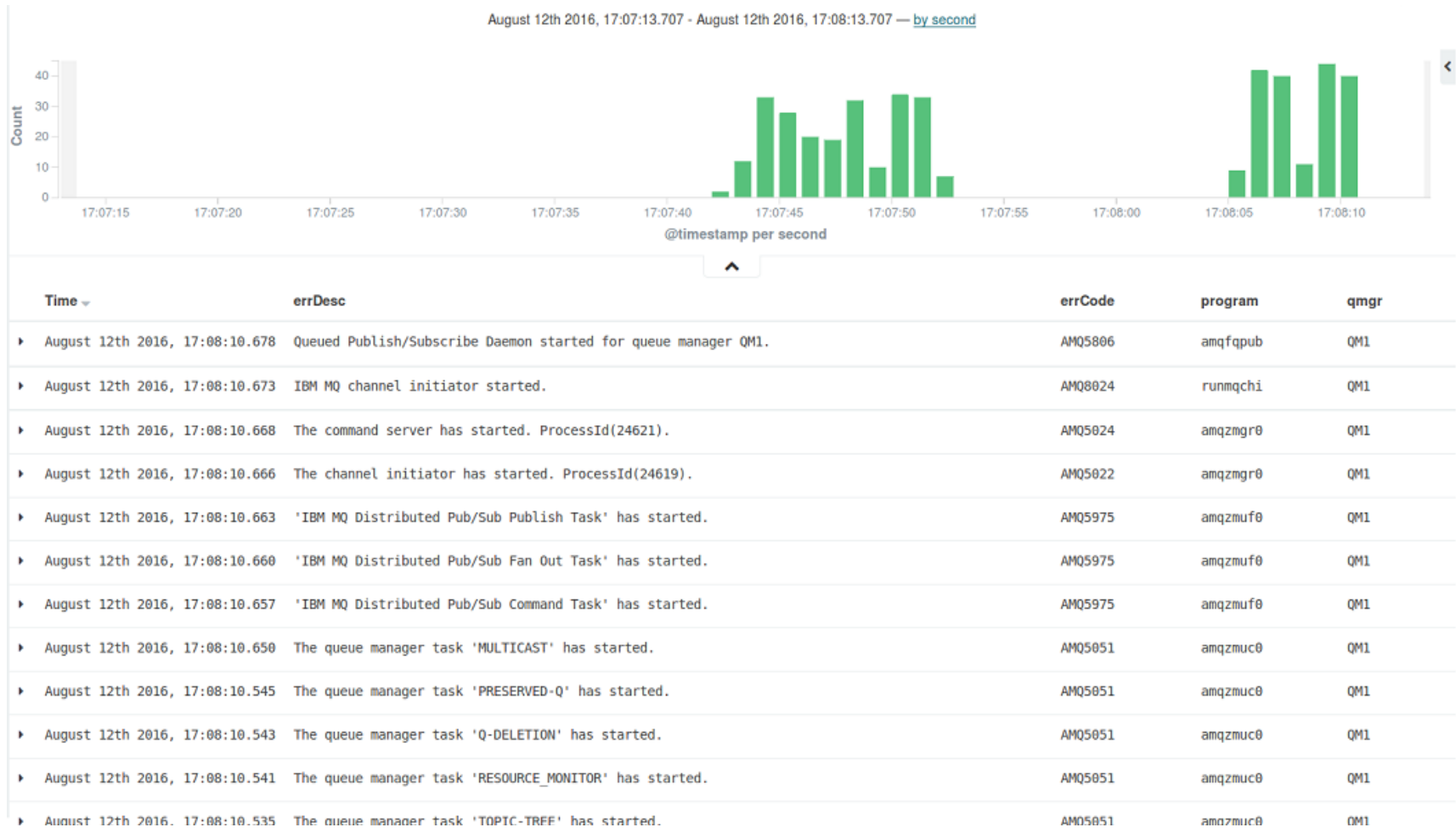
Time August 15th 2016, 09:45:53.403

message: 15/08/16 09:45:52 - Process(26913.1) User(mwhitehead) Program(runmqchl) Host(mrw-ubuntu-1604) Installation(Installation2) VRMF(9.0.0.0) QMgr(QM1) AMQ9002: Channel 'QM2' is starting. EXPLANATION: Channel 'QM2' is starting. ACTION: None. @version: 1 @timestamp: August 15th 2016, 09:45:53.403 path: /var/mqm/qmgrs/QM1/errors/AMQERR01.LOG host: mrw-ubuntu-1604 type: amqerrlogs tags: multiline date: 15/08/16 time: 09:45:52 process: 26913.1 user: mwhitehead program: runmqchl hostname: mrw-ubuntu-1604 installation: Installation2 vmrf: 9.0.0.0 qmgr: QM1 errCode: AMQ9002 errDesc: Channel 'QM2' is starting. explanation: Channel 'QM2' is starting. id: AVaNY0q1G7XHmVRAy34 type: %[@metadata][type] index: %[@metadata][type]-2016.08.15/%[@metadata][type]/AVaNY0q1G7XHmVRAy34

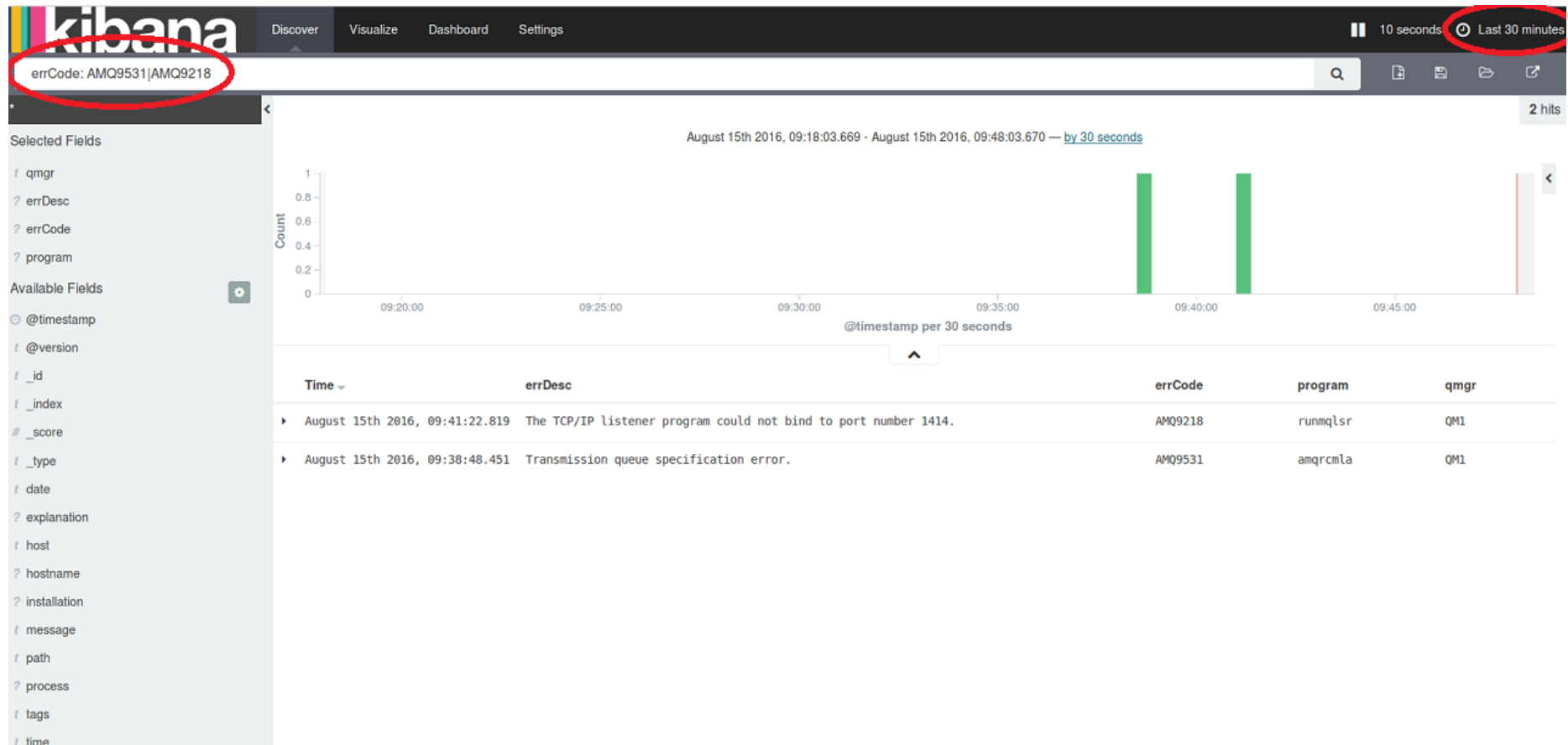
Table JSON

@timestamp	August 15th 2016, 09:45:53.403
@version	1
_id	AVaNY0q1G7XHmVRAy34
_index	%[@metadata][type]-2016.08.15
_score	
_type	%[@metadata][type]
date	15/08/16
errCode	AMQ9002
errDesc	Channel 'QM2' is starting.
explanation	Channel 'QM2' is starting.
host	mrw-ubuntu-1604
hostname	mrw-ubuntu-1604
installation	Installation2
message	15/08/16 09:45:52 - Process(26913.1) User(mwhitehead) Program(runmqchl) Host(mrw-ubuntu-1604) Installation(Installation2) VRMF(9.0.0.0) QMgr(QM1) AMQ9002: Channel 'QM2' is starting. EXPLANATION: Channel 'QM2' is starting. ACTION: None.
path	/var/mqm/qmgrs/QM1/errors/AMQERR01.LOG
process	26913.1
program	runmqchl
qmgr	QM1
tags	multiline
time	09:45:52

Kibana Dashboards



Kibana Dashboards



Kibana - Summary

- You can add and remove different components to the dashboard
- You can save dashboards once you're happy with their layout
- Create various charts and graphs of your data.

AWS CloudWatch

- AWS CloudWatch is a monitoring & management service
- Provides actionable insights to monitor applications
- Respond to system-wide performance changes
- Optimize resource utilization
- Collects monitoring & operational data

AWS CloudWatch



Amazon CloudWatch

Complete visibility into your cloud resources and applications



Collect

Metrics and logs from all your AWS resources, applications, and services that run on AWS and on-premises servers



Monitor

Visualize applications and infrastructure with CloudWatch dashboards; correlate logs and metrics side by side to troubleshoot and set alerts with CloudWatch Alarms



Act

Automate response to operational changes with CloudWatch Events and Auto Scaling



Analyze

Up to 1-second metrics, extended data retention (15 months), and real-time analysis with CloudWatch Metric Math



Application Monitoring



System-wide Visibility



Resource Optimization



Unified Operational Health

AWS CloudWatch

PROS	CONS
Feature rich	Less powerful for non AWS resources
Easy to setup	Need work for customization
Can search individual log streams	Not as advanced as ElasticSearch
Can setup filters and alarms	

AWS CloudWatch

- Make sure that your EC2 instances are authorized to talk to the CloudWatch service.
 - Create a policy in the Identity and Access Management (IAM) service, and then assigning that policy to a role.
- Roles can then be assigned to your EC2 instances.

Sending error logs to CloudWatch

- CloudWatch requires a CloudWatch agent to be installed on your EC2 instance.
- This is a small Python program which will monitor log files, identify separate log entries, and then send those entries to CloudWatch.
- After installing the CloudWatch agent, you can add the necessary configuration for the MQ error logs.

Sending error logs to CloudWatch

- This configuration can either be written to the main CloudWatch configuration file in `/var/awslogs/etc/awslogs.conf`, or to a standalone file in `/var/awslogs/etc/config/mq.conf`.
- The configuration identifies two files to watch: the main MQ error log, and the error log for queue manager.
- It configures the date/time format, tells the agent to send any pending log messages to CloudWatch at least every five seconds, and write them to a "log stream" based on the EC2 instance ID.
- Restart CloudWatch service and view the logs in the AWS management console (under CloudWatch -> Logs)

CloudWatch – Summary

- CloudWatch allows you can search individual log streams (per instance), and set up filters and alarms for certain events.
- The search facilities offered aren't anywhere near as advanced as ElasticSearch, but you can choose to forward log events on to ElasticSearch if you want to.
- Importantly, you can also trigger AWS Lambda functions based on certain messages being logged, which will allow you to take programmatic action.
- You can use either solutions and also add email functionality to automatically indicate when specific errors occurs.

Prometheus

- An open-source monitoring and alerting solution, whose particular strengths:
 - the collection of time series data, with the ability to easily query that data.
 - it provides libraries in several languages to enable products to export data to it.
 - it works by pulling information from exporters such as this MQ program at configured intervals over an HTTP connection.

Prometheus

- The Prometheus documentation has information on more complex configuration options.
- Once the Prometheus server has picked up the MQ configuration, the metrics can be seen under the specified jobname.

Grafana

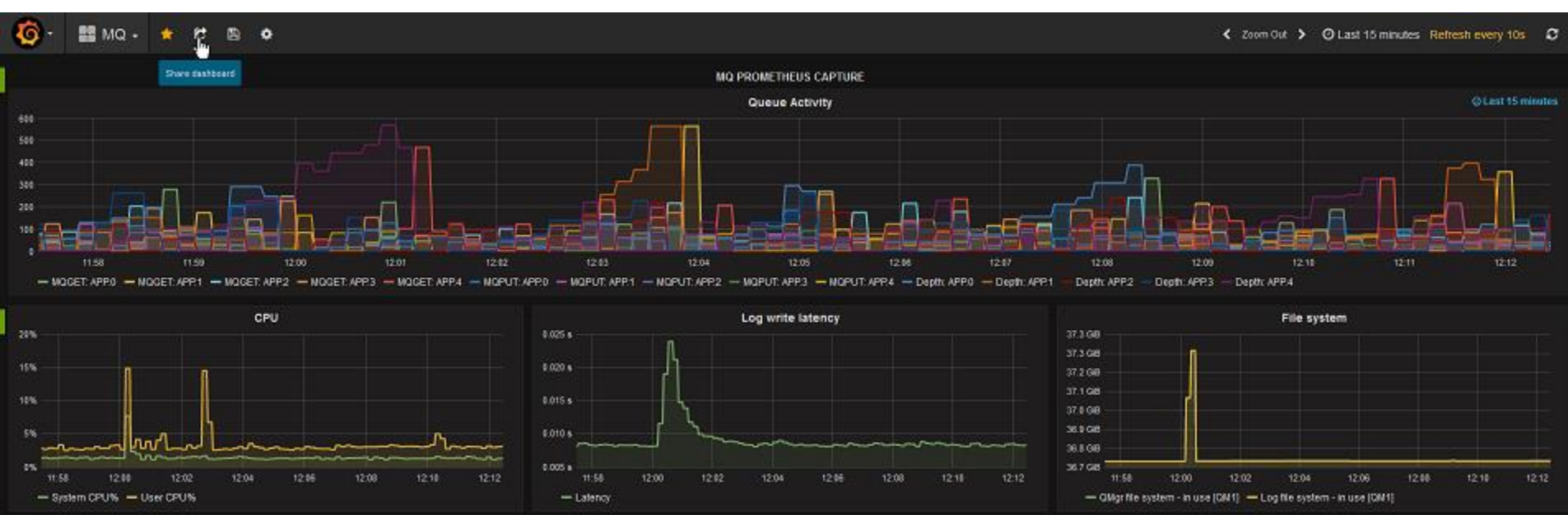
- Grafana provides a way to create dashboards and visualize data held in time series databases.
- It has Prometheus as a built-in data source.
- For example, the number of MQPUTs to a queue may be of interest, and this kind of database makes it easy to see how many operations occurred in an interval, or calculate averages.

Configuring Grafana

- Use Prometheus (only a hostname and portnumber is required when adding this type of datasource.)
- All of the MQ metrics can be accessed and added to dashboards.

Configuring Grafana Dashboards

As an example, this dashboard is looking at several items including the same queues as above, and CPU and logging information:



Grafana Dashboards

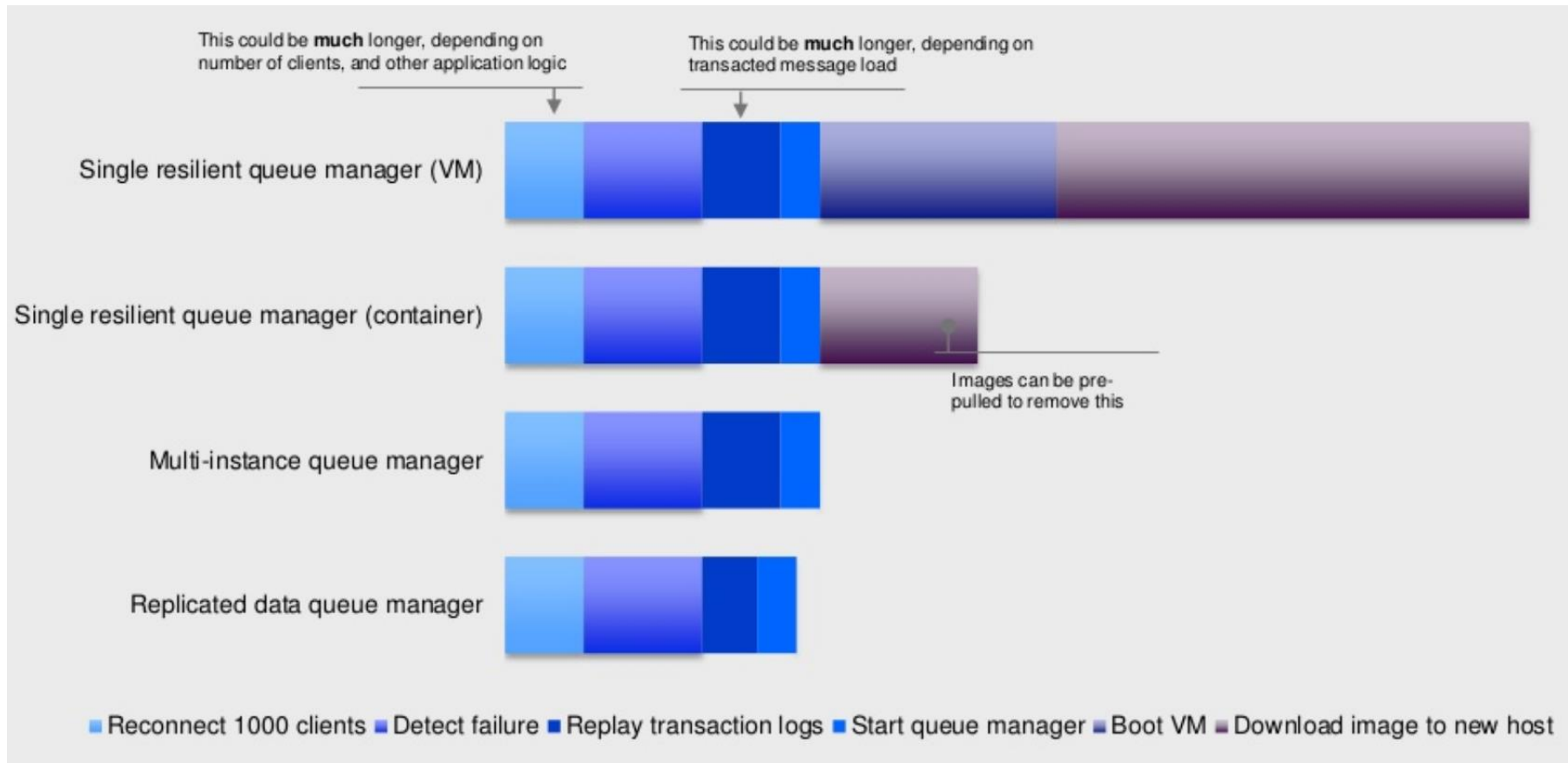
This picture shows how the top panel was configured, to select several metrics and show the object name in the legend

The screenshot shows the Grafana dashboard configuration for the top panel. The panel is titled "Graph" and has tabs for General, Metrics, Axes, Legend, Display, and Time range. The Metrics tab is selected. There are three queries listed, each with a query name, a legend format, and a metric lookup.

Query	Legend format	Metric lookup
ibmmq_object_mqget	MQGET: {{object}}	ibmmq_queue_messages_de
ibmmq_object_mqput_mqput1	MQPUT: {{object}}	ibmmq_queue_messages_er
ibmmq_object_queue_depth	Depth: {{object}}	metric name

At the bottom, there is a "Panel data source" dropdown set to "MQ Prometheus" and a "+ Add query" button.

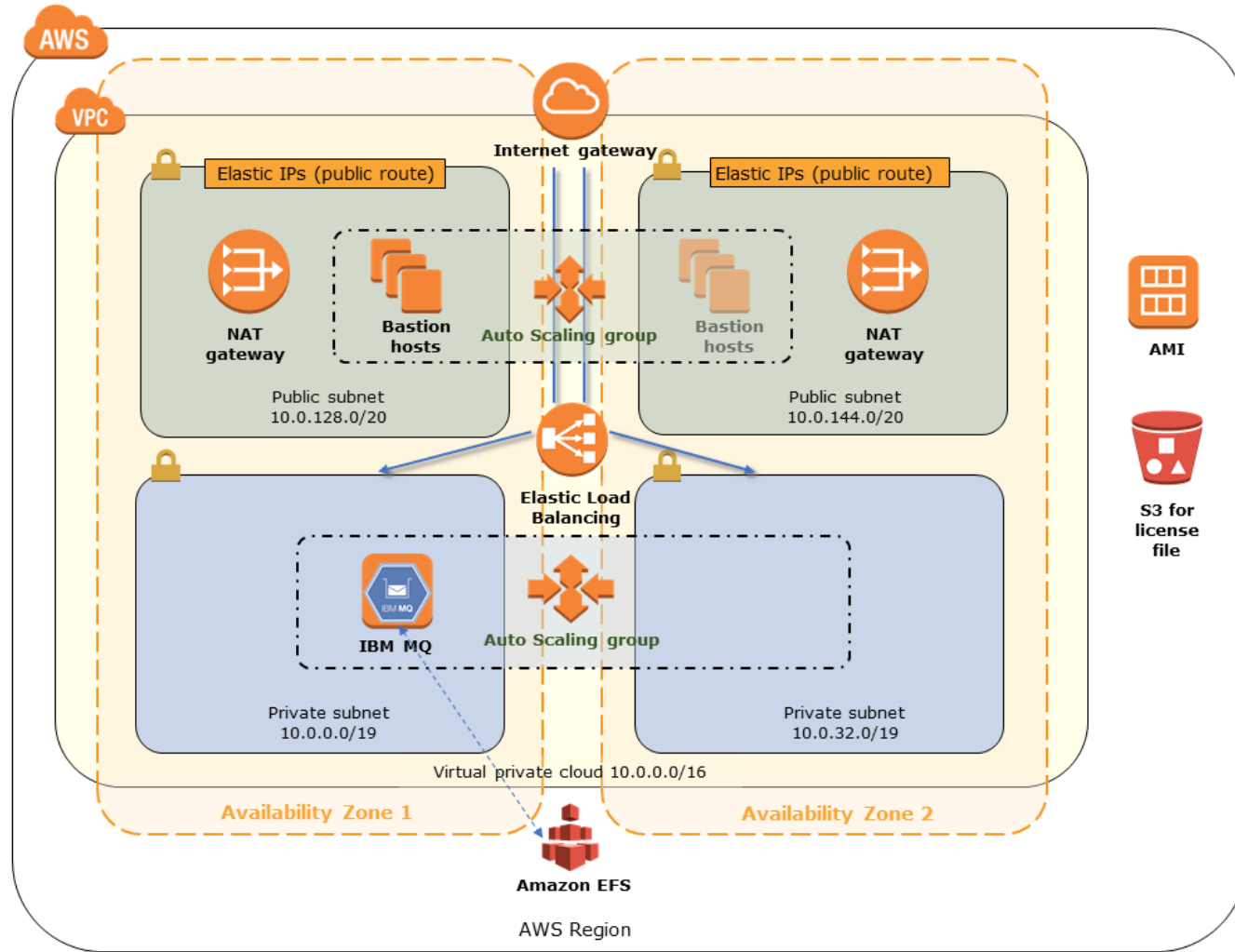
MQ Recover times (*estimates*)



MQ on AWS Options

- Quick Start Guide for AWS (MQV9)
- Partner World Licensing (non-prod environments)
- BYOL from your current license stack
- Purchase of MQ from AWS Marketplace (MQV8)
- MQ Clients on AWS (connecting back to your on-prem qmgrs) – no software license costs!

MQ on AWS (the quick start with trials)



MQ from the AWS MarketPlace

MQ from the AWS Marketplace

This AMI has WebSphere MQ v 8.0 installed on Red Hat Enterprise Linux 7.5 base.

Highlights:

- Install WebSphere MQ in just a few minutes
- Ideal for Development, Test and peak production workloads
- If the version you require is not listed, you can request it.

COSTS

Hourly Cost:

EC2 Instance type	Software/hr	EC2/hr	Total/hr
<input checked="" type="radio"/> m3.medium ★ Vendor Recommended	\$0.8	\$0.127	\$0.927

Annual Cost:

EC2 Instance type	Software/yr	EC2/hr	Percent Savings (%)
<input checked="" type="radio"/> m3.medium ★ Vendor Recommended	\$5,957	\$0.127	15%

AWS Free Tier

12 months free and always free products

AWS Free Tier includes offers that expire 12 months following sign up and others that never expire.

Amazon EC2

750 hours per month of Linux, RHEL, or SLES t2.micro instance usage

750 hours per month of Windows t2.micro instance usage

Amazon QuickSight

1 GB of SPICE capacity 1 user perpetual free tier

10GB of SPICE capacity the first 2 months for free for a total of 4 users

Amazon CloudWatch

10 Custom Metrics and 10 Alarms

1,000,000 API Requests

5GB of Log Data Ingestion and 5GB of Log Data Archive

3 Dashboards with up to 50 Metrics Each per Month

Amazon RDS

750 Hours per month of db.t2.micro database usage (applicable DB engines)

20 GB of General Purpose (SSD) database storage

20 GB of storage for database backups and DB Snapshots

Amazon S3

5 GB of Standard Storage

20,000 Get Requests

2,000 Put Requests

AWS Lambda

1,000,000 free requests per month

Up to 3.2 million seconds of compute time per month

Summary

- These slides and presented material show how the statistics generated by MQ can easily be used in some of the monitoring packages that are commonly used with various cloud and container-based systems.
- The MQ data can be integrated with other metrics to give a complete view of your environment.

Useful Links

- MQ on Cloud – IBM Messaging

<https://developer.ibm.com/messaging/mq-on-cloud/>

- GitHub Repositories

<https://github.com/ibm-messaging/mq-aws>

- IBM MQ - Using AWS CloudWatch to monitor queue managers

https://www.ibm.com/developerworks/community/blogs/messaging/entry/IBM_MQ_Using_CloudWatch_to_monitor_queue_managers?lang=en

- IBM MQ - Using Prometheus & Grafana to monitor qmgrs

https://www.ibm.com/developerworks/community/blogs/messaging/entry/IBM_MQ_Using_Prometheus_and_Grafana_to_monitor_queue_managers?lang=en

Useful Links

- Storing and searching MQ error logs in Elasticsearch
https://www.ibm.com/developerworks/community/blogs/messaging/entry/Storing_and_searching_MQ_logs_in_Elasticsearch?lang=en
- MQ on AWS: Sending MQ error logs to CloudWatch
https://www.ibm.com/developerworks/community/blogs/messaging/entry/mq_aws_cloudwatch_logs?lang=en

Grafana Support For Prometheus

<https://prometheus.io/docs/visualization/grafana/>

Questions & Answers

