**Red Hat – Operator Enablement – Problem Solving Questions**

Approach / Brainstorm Methodology:

Have been presented with two questions (both problems are in areas of experience) and want solid responses completed before start of the week 3/1. Generally this would have taken 5 days, but want to come in ahead of that with a best effort.

The start time will be late Tuesday 2/23, with some general background / high level exploration. The bulk of the work should be sorted out on 2/24 and 2/25. By Friday morning, would like to be at the point to be polishing / reviewing and then turn in close to COB for the week.

**The first question will require a presentation and demo of a web app:**

- Initial notes: 2/23

- For the presentation, would like to have a power point slides since there may be areas where there would be some explanation and some diagrams

- See Q1 sub questions / topics for discussion

- Cover the questions but do NOT limit yourself to the topics

- Will need to setup files for the containers: webapp using Django (Python) and MySQL

- need to sort out between having spate containers (prod like) vs.

single container (dev portable)

- for ease, will probably use a single docker-compose for building the containers separately

- Would like to have this webap in my AWS environment with RHEL 8 using podman

- Fallback would be RHEL / Centos 7 using docker

- Would prefer NOT to use my local VirtaulBox RHEL or Ubuntu setups

- Will need to list references / urls

- Have worked with Docker containers and images before, however, these will both be new to me.

- Have installed and setup MySQL before, just not in a container. Based on experience with Oracle containers, the tricky part will be the setup of the user and permissions on the shared space for the db files. Should be easily able to find a MySQL container out there setup a simple db to use for the sample webapp, save off the image, then reload it. Probably good idea to do the DB portion first, since I am more familiar, and the web app will need this interaction… so good idea to have this in place first. Perhaps wait to create the new container until the web app can communicate with the initial running image.

- As for Django, I have not used this before (have installed and used apache/Tomcat / WebSphere on VMs). I do know some Python, and follows the same form as C++ (being OO , similar syntax, and using class imports) but more like Javascript so that there is no compile step needed. The trick to this will be to find a decent Django container to use, write a sample program that interacts with the DB. Will probably need to allocate more time to this portion of the question.

- Since I am planning on using AWS for development of the sample app, will plan to use RHEL 8, which will lead down the path of leveraging podman and related tools for containerization. I have not had a great deal of experience with podman, but feel very confident that there should be plenty Red Hat docs and example out there to reference. Best case is there is an existing lab that already incorporates

- Would be nice to have files checked into my Git repo for consumption / demo purposes.

- Once the basic web app is working, would also be a good idea to take a video / screen shots before starting on the containerization of everything (just in case things start breaking / cannot revive properly).

- Was able to setup AWS RHEL 8 vm setup with podman and container tools first. Try find some basic containers to load for a bit until I reach enough blockers to shift gears to take a better dive into the questions to address. This demo will be interesting with setting up AWS-EC2-RHEL8 with podman creating a web app… hopefully will not hit too many snags along the way, but with all linux apps, there will be some permissions / network issues to sort out. It will be very tempting to keep going back to sorting out the containers, but probably a good idea to push through the written questions then circle back.

- Did a decent amount of setup for the VM in order to get setup for using containers, and user setup for possible interaction with MySQL. This is documented in my brain dump notes.

- Did some google searching on ‘msql django docker compose’ found a good deal of great resources and will leverage some tutorials and github repos in order to put together a simple sample app using **podman-compose**.

- Did some searching on you tube for ‘msql django docker compose’ and found some helpful tutorials that combined with some sample code from google searches, should be able to have a decent sample web app running. To note, it seems that pairing django with postgres is a very common approach / combo. I have used both postres and mysql in different ways, and I do find postgres a bit easier to setup.

- Update: 2/24

- Have had good success with keeping true to using RHEL8 and using podman. Found a nice tool that is the equivalent to making docker-compose work with podman as podman-compose.

Reference: <https://github.com/containers/podman-compose>

Since there are lots of examples / tutorials using docker-compose, this helped stay the course very nice.

- One point of frustration was that I REALLY wanted to try to keep everything as Red Hat centric as possible, and that included using a RHEL8 MySQL8 published docker container image from here: <https://catalog.redhat.com/software/containers/rhel8/mysql-80/5ba0ad4cdd19c70b45cbf48c?container-tabs=overview&gti-tabs=registry-tokens>

I was attempting to learn more about deploying with mysql docker container file, wince the web app would depend on mysql, thought it would be good to explore. When trying to run:

podman run -d --name mysql\_database -e MYSQL\_USER=mysql -e MYSQL\_PASSWORD=mysql4fun -e MYSQL\_DATABASE=db -p 3306:3306 rhel8/mysql-80

There were no issues, spinning up the image and logging into the container. However, I knew that the shared volume would need to be specified. So tried with this (recommended in the Red Hat doc).

podman run -d --name mysql\_database -e MYSQL\_USER=mysql -e MYSQL\_PASSWORD=mysql4fun -e MYSQL\_DATABASE=db -p 3306:3306 rhel8/mysql-80 -v /mysql/data:/var/lib/mysql/data

This returned a dead on arrival container, so I figured it was perms (which the doc and my linux sense) led me to think was an issue. So I had to remove / deploy the container to get a running container without the volume, and ensure the GID, UID, User Names, and file permissions (a few times). Setup the local user on the VM with the same GID, UID and set the file perms accordingly, and still had issues. After coming back to the issue several times, I checked the podman log to find returned error:

podman logs -f mysql\_database

/usr/bin/container-entrypoint: line 2: exec: -v: invalid option

exec: usage: exec [-cl] [-a name] [command [arguments ...]] [redirection ...]

After going back and forth a few more times, I revisited the dockerfile:

<https://catalog.redhat.com/software/containers/rhel8/mysql-80/5ba0ad4cdd19c70b45cbf48c?container-tabs=dockerfile>

and found:

# Not using VOLUME statement since it's not working in OpenShift Online:

# https://github.com/sclorg/httpd-container/issues/30

# VOLUME ["/var/lib/mysql/data"]

Then with some google searching, found this error in fact is not resolved.

I tried a few other methods to try to brute force adding a shared space after container deploy and came up short. So unfortunately, had to cut my losses and ditch the RHEL8 MySQL8 container.

- Update: 2/25 – Clarification Response from Matt:

> For Q1, I am working on a PPT for the partner scenario and am under the assumption that the current architecture is using multiple VMs instead of a single VM, however, for the response, does this really difference in the proposed approach ?

**Let's assume the current architecture is using two vm's: one frontend vm, and one backend vm.**

> For clarification, you want to see a working demo and not just the setup files ?

**Whatever you feel like presenting - files are fine**.

**Provide your reasoning for setting up django and mysql in the same container.**

> I thought a PPT for this response would not come across with much clarity.  Is this a good approach ?

**Certainly!!**

- In working to address all questions, have found there is a lot of verbiage on the slides, trying to trim down and rephrase to save space.

- Somewhat torn where to discuss build files and demo, and ultimately have decided to put at the end of the demo. Had originally put after the arch slide (VMs vs Containers), since sometimes it is nice to have a change of pace between slides. However, there could be more discussion with the files / demo and may make more sense to complete all slides first.

- Have decided to split out the demo files to have separate containers for the Django web and MySQL Db. Had originally put them in the same container, which I do think it lends itself to be a more elegant approach and is very useful for developers, but not a great idea for production. To keep a good flow for Partner X moving towards production containers, it would be better to split them out now. This lends itself to using the web app container in multiple instances for fail over if using a load balancer. Having two db’s writing to the same volume would not be a good. Will still plan on creating a docker compose with a db and a web app for the demo. Will note if more web apps are to be started for loading balancing in a production environment, then this should be able to altered on the command line when evoking the podman-compose up.

- Update 2/26: Was expecting to have these questions completed last night, but ended up spending more time answering the Q1 questions about docker. Had to do some double checking with web searching. Still have an abundance of verbiage, but most slides are looking good.

Created another AWS-EC2-RHEL8 vm to do more container testing. Have a better approach for setting up the demo, and this should help finalize some of my open ideas in the slides. My goal is to have the demo working soon (or at least very close). The last items on the todos is to clean up the PPT, wrap up the demo, and put in comments in the build files, then send all the related docs to Matt and John.

- Hit another issue with the setup of the Django web server:

In the Dockerfile for the app, I have a copy statement, that is possibly not needed, but going forth with Django and having the files in a repo, the app dir will hold the python files for the sample app. Similarly, when I have the setup for the msql db, will be using a db folder.

Dockerfile snipit (for full notes see: rh\_prob\_solve\_brain\_dump.txt)

- Ran into too many issues with using podman with podman-compose. Will shift to using Dock on EC2-AMI

- Update 2/28 – Stepped away from the questions over the weekend, and was able to make good use of time to polish up the PPT. Sorted out environment issues with EC2-AMI. Turns out, getting a decent python 3 / Django environment can get messy quick on AMI. Used a work around with setting up a virtual environment with the latest python, Django, pip, and related tools. Turns out, AWS does not like to have any sym links or alternatives changed… otherwise, it can become a challenge to get back to working order. Have some good container files to review, and the containers run, but is very basic.

Finalized the presentation for Q1 and have some build files in place. Spent a good deal of time sorting out containers to use and using podman. Ultimately found a few a few references that are helpful to sort out the q1 /demo. Need to focus on Q2, finish an arch diagram and explanation of services and usage.

**The second question will require detailed description based on AWS services and how to leverage usages of the AWS services for a partner.**

- Was just doing some AWS training for certification course, so this is fresh in my mind (hooray). - This may not lend itself well to a power point, but still would like to have some organization in tables / bullet points for ease of readability / presentation / discussion.

- The second question will need to use description details that the partner can understand then relate to their usage. Try not to bore the audience with dry cut/paste definitions.

- Good idea to add a few more services of note that may be helpful for the partner

- Will need to list references and case studies

- May go ahead and make headway on Q2 first since AWS training is still pretty fresh, then work between Q2 and Q1, with Q1 probably taking the bulk of time.

- This should include a diagram to help explain the AWS services. Found an AWESOME online arch diagram app that includes a library of AWS symbols. Signed up for a free account at: <https://app.creately.com>

- Update: 2/25 - Made good progress with Q2, there is still a lot of info to sift through and cut down, and keep the tie in with how the partner will use each AWS service. Included a few more services that I thought should be added to the architecture for the streaming service. Spent some good time on the arch diagram (working on a diagram was a lot of fun with this new tool) which has always been a staple in any good architecture story.

- I have been reviewing this material, and feel that working between the questions, demo, and arch diagram has broken up the process so I do not get too bogged down in any one topic at a time, thus am able to keep fresh. From previous experience with working on arch docs, education, and presentations for customers, another team member’s review on the material will be of great help, which is probably what the review will be like once these answers are turned in. I know there is a good amount of information to cover and I am sure that I have missed a few items in the description write up and arch diagram.

- Update: 2/26 – The arch doc and questions are in great shape. I think this is a solid arch doc. It includes some nice commentary and a very nice arch diagram (REALLY pleased with this arch diagram and finding the online site to develop it). The responses look good in the tables and am glad I spent some time to clean up the formatting of the table layout. Made sure to incorporate adding in ‘Partner X’ as much as possible and how they would use the AWS services. Included reference links, some case studies for Partner X to review. Closed the arch doc with a note about items to consider when working with AWS. I am pleased with how this turned out (did I mention that I was excited about the arch diagram).

**Question 1: Listed here for reference notes**

You are assisting a Red Hat Partner with building a solution for their Python-based web application.

The web application uses Django for the front-end UI and MySQL for the backend database.

The application is currently hosted on virtual machines but they are interested in migrating to containers.

You have been instructed to assist the Partner with understanding the advantages (and disadvantages) of containers compared to their current virtual machine solution. They also want you to present a demo of this containerized solution with a sample web application using both Django and MySQL.

The web application can be extremely basic and is used for a simple demo of the containerized web application.

Consider the following:

* What will the container build files for the front-end and back-end images look like?

Provide an example of each.

* How would we ensure the containers run as non-root users?

Why is that important?

* How do we ensure the customer’s data is persistent, backed-up, and restored?

Why is that important?

* What does networking look like in this scenario?
* How does communication occur between the front and backend services?
* Will the customer need to make any adjustments to firewall rules?
* Where will the container images reside and how will the containers access images in a private repository?
* How can we handle single-points of failure in a containerized environment?

Are there any existing tools or software that can help with this?

**Question 2: Listed here for reference notes**

A Red Hat Partner is currently using Amazon Web Services (AWS) for their music streaming app.

The application allows users to:

- Search for music based on: artist, album, or genre

- Allows them to create, edit, and share playlists

- Describe each AWS service listed below

- How the Partner would use that service to **architect** their application.

- Give as much **detail** as possible.

\* EC2

\* EC2 Autoscaling

\* Elastic Load Balancing

\* VPC

\* Route 53

\* S3

\* CloudFront

\* DynamoDB

\* Simple Notification Service (SNS)