Project:
Plan, Reduce, Repeat

Hui Ren:

June 11, 2022

Overview:

You have recently joined the SRE team for an exotic plant reseller startup. They already have a small SRE team in place consisting of two other members. You are just finishing up your training period and are now ready to be on your own.

You have a busy week ahead of you as there is a release this week plus your on-call shift. Part of your release duties includes helping to maintain the as-built document by adding this new release, as well as planning for system resource changes. For your on-call shift, you have to respond to alerts as they come in and write up an on-call summary to document your shift. Finally, you'll round out your week by helping to reduce toil. You will have to identify any toil you encounter throughout the week and create a toil reduction plan. After you have a plan all ready, you will need to work on implementing that plan by writing some scripts to help automate tasks.

Scenario 1

Release Day

Release Night

Summary

Tonight is release night, and it will be your first time assisting with a release as an SRE. The process now is manual, with no real consideration for how releases may impact resource allocation. Luckily, your other team members have started implementing an as-built document. You'll have to add tonight's release to the document. The release is a pretty major release with the addition of a new feature that will bring in a large number of new clients. Looking at the results from testing, you can see that this new feature is going to add additional resource requirements as it is both more memory and, to a lesser extent, CPU intensive than before.

Current Release Features

This release will have the following changes that will need to be documented on the as-built design document. The developers have been hard at work implementing the following tickets:

- Ticket 203 added a new catalog for exotic plants. This ticket added new tables in the database to handle the additional catalogs.
- Ticket 202 rearranged the catalog menu in the UI to accommodate the additional catalog, as well as making it more user-friendly.
- Ticket 201 added an additional component to the application, an order processor. The order processor is responsible for batch processing orders on a schedule. The reasoning behind this was to decouple the UI from order processing, and since order processing can be CPU intensive, this decoupling prevents the app from performing poorly. The Design Doc 5247 goes into more detail about the design specifics.
- Ticket 205 fixed a security flaw where attackers could execute a SQL injection attack.

Release Night, cont.

Release Process

The established release process is a manual affair generally done by one of the operations team members. The OPs team generally will download the latest code, shut down the app, run the database migrations, change or add any needed configurations and then start the app back up. In the past this has caused issues as steps have been forgotten, not all the scripts were executed, the app was not restarted properly, among other issues. During the release window, the OPs engineer would also add new resources as needed. This has led to downtime in the past as the app became overloaded and could not serve requests anymore.

Release Planning

During load testing for this release, it was determined that

- Main Application
  - The new catalog feature increases RAM usages by 25% for the same number of users, while not increasing CPU significantly. Currently, the main application containers are utilizing almost 85% of the RAM allocated.
  - At the current resource allocation, each server can handle 500 concurrent users. Currently, there are 3 application containers to support about 2000 total users, with about 1300 being on at any one time. This release is expected to add about 1.5 to 2.5 times the total number of users, with a need to handle 2600 users concurrently.
- Order Processor
  - This component has a high CPU utilization with moderate RAM requirements. In testing, a full loaded queue used approximately 1 Gb of RAM.
  - The component runs with 2 concurrent processes, pulling orders out of the database and processing them for fulfillment. This component can process 4 orders at a time, with the average order taking between 10 and 15 seconds to complete depending on the size and complexity of the order as well as CPU resource allocation. QA recommends twice the CPU as the main application.
- Database
  - The database was provisioned to handle a much larger application than what the company has now and passed the load tests with flying colors.

As-Built Doc Template
Release Version

Stakeholders

These are the teams and members involved in this reason. This should include ops members, developers, SRE members, database admin, etc

Code Changes

This section should include a list of code changes going into this release separated into groups (for example, by bug fix, feature addition, and security fixes). This should be a short summary of the change with a ticket included to follow up with for more detailed information.

Data and System Changes

This should be formatted similarly to the code changes section, except listing any changes to the data model (database or API changes) or system changes.

Design decision highlights

Document the high-level reasoning behind any design choices. This section should only include a summary of the design decision with links to supporting documentation to follow up with for more detailed information.

Test Section

In this section, list any notable highlights from testing. Things to include here would be any changes to the testing methodology, changes to the test performed, and any tests that are not currently pass (or pass with a warning).

Deployment Notes

Include any changes made to the deployment process or any changes that should be made to improve in feature releases.

As-Built Doc
Release 1

Stakeholders

- Developers
  - John Doe
  - Jane Peters
  - Sam Ross
- Ops
  - Jay Smith
- SRE
  - John Robert

Code Changes

- Security fixes
  - Added new password requirements (Tk-100)
  - Fixed how SQL queries were handled (Tk-103)
- Feature Additions
  - Added new menu options for users (Tk-102)
  - Users can now have middle names (Tk-101)

Data and System Changes

- Data model changes
  - Added columns for middle names in user table (TK-101)
  - Added additional New Menu table (Tk-102)
  - Users table was split into 2 smaller tables (TK-101)
  -

As-Built Doc
Release 1

Design decision highlights

Users table was split into two smaller tables to create more efficient queries and mappings. Keeping it as one big table began to cause slow queries and allowed for a larger number of users. See Design Doc 134 for further discussion.

Test Section

All test suites are passing 100%.

Deployment Notes

The database admins asked for an additional set of scripts to be run for data corrections.

Deployment File Release 1

```
ApiVersion: apps/v1
kind: Deployment
metadata:
  name: app-deployment
  namespace: prod
  labels:
    app: mainApp
spec:
  replicas: 3
  selector:
    matchLabels:
      app: mainApp
  template:
    metadata:
      labels:
        app: mainApp
    spec:
      containers:
      - name: mainApp
        image: nginx:latest
        resources:
          requests:
            memory: 256mb
            cpu: 250m
        ports:
        - containerPort: 80
```

As-Built Doc
Release 2

Stakeholders

- Developers: John Doe, Jane Peters, Sam Ross
- Ops: Jay Smith
- Database Admin: John Robert
- SRE: Hui Ren

Code Changes

- Security fixes
    - Fixed a security flaw where attackers could execute a SQL injection attack (Tk-205).
- Feature Additions
    - Added a new catalog for exotic plants (Tk-203).
    - Rearranged the catalog menu in the UI to accommodate the additional catalog, as well as making it more user-friendly (Tk-202).
    - Added an additional component to the application, an order processor. The order processor is responsible for batch processing orders on a schedule.
    -

As-Built Doc
Release 2

Data and System Changes

- Data model changes
  - Added new tables in the database to handle the additional catalogs (Tk-203).
- System changes
  - For mainApp, increase cpu to 320mb.
  - For the order process, allocate 1024mb memory and 512mb cpu.

Design decision highlights

The application was split into the main application and an order processor. The reasoning behind this was to decouple the UI from order processing, and since order processing can be CPU intensive, this decoupling prevents the app from performing poorly. The Design Doc 5247 goes into more detail about the design specifics.

Test Section

All test suites are passing 100%.

Deployment Notes

Use terraform to deploy infrastructure.

Use blue-green deployment if zero downtime is required.

Deployment File Release 2

Update the file for Release 2 to match the description in the scenario:

```
ApiVersion: apps/v1
kind: Deployment
metadata:
  name: app-deployment
  namespace: prod
  labels:
    app: mainApp
spec:
  Replicas: 7
  selector:
    matchLabels:
      app: mainApp
  template:
    metadata:
      labels:
        app: mainApp
    spec:
      containers:
      - name: mainApp
        image: nginx:latest
        resources:
          requests:
            memory: 256mb
            Cpu: 320mb
        ports:
        - containerPort: 80
      - name: order_processor
        image: nginx:latest
        resources:
          requests:
            Memory: 1024mb
            Cpu: 512mb
        ports:
        - containerPort: 80
```

Scenario 2

On-Call Shift

On-Call Shift

Summary

Today is your first on-call shift as an SRE. During your shift, you will have to respond to alerts to keep the system running at its best using the on-call best practices learned in this course. During your on-call shift, make sure to be thinking of ways to reduce toil. After your on-call shift is over, you will be responsible for writing a summary of your shift and a post-mortem. On the following slides you will encounter several different “alerts” from your monitoring stack. Each “alert” will contain several different parts that will help you write your on-call log for your shift. Additionally, you’ll encounter an application outage that will require a post-mortem.

Alert Components

Summary -- This will be general knowledge about the systems involved that you would know if you had actually been working at the company. It will include a brief description of the systems involved as well information about how it is managed.

Standard Operating Procedure (SOP) -- This will be a short description of the steps to troubleshoot and potentially correct the cause of the alert.

Log and Monitoring Details -- This section will contain snippets of relevant logs and monitoring data (graphs, metrics, etc.) that are associated with responding to an alert.

On-call Log

After your on-call shift you’ll need to add to the on-call log. There is a provided sample template for you to use that includes all the necessary fields. Remember your on-call log is used to help track recurring alerts/issues as well as providing a record of the steps taken to resolve the issue.

Post-Mortem

Unfortunately there will be an application outage on your shift that will require a post-mortem. You will only be responsible for filling in your involvement, plus you’ll be in charge of creating an action plan and impact assessment.

On-Call Shift -- Alert 1

Order Processing Issues

Summary

You receive an alert that the number of Outstanding Orders is too high. Orders are processed by a separate component from the main application. It runs periodically (every hour currently) to batch process any open orders. Your team has set up some monitors to keep track of how well the order processor is doing.

SOP

Number of Outstanding Orders is Too High

If this alert comes through you will need to check the dashboard to see if the Order Processor is overloaded with orders. If there is a high number of orders contact Ops to see if the processor should be run more frequently.

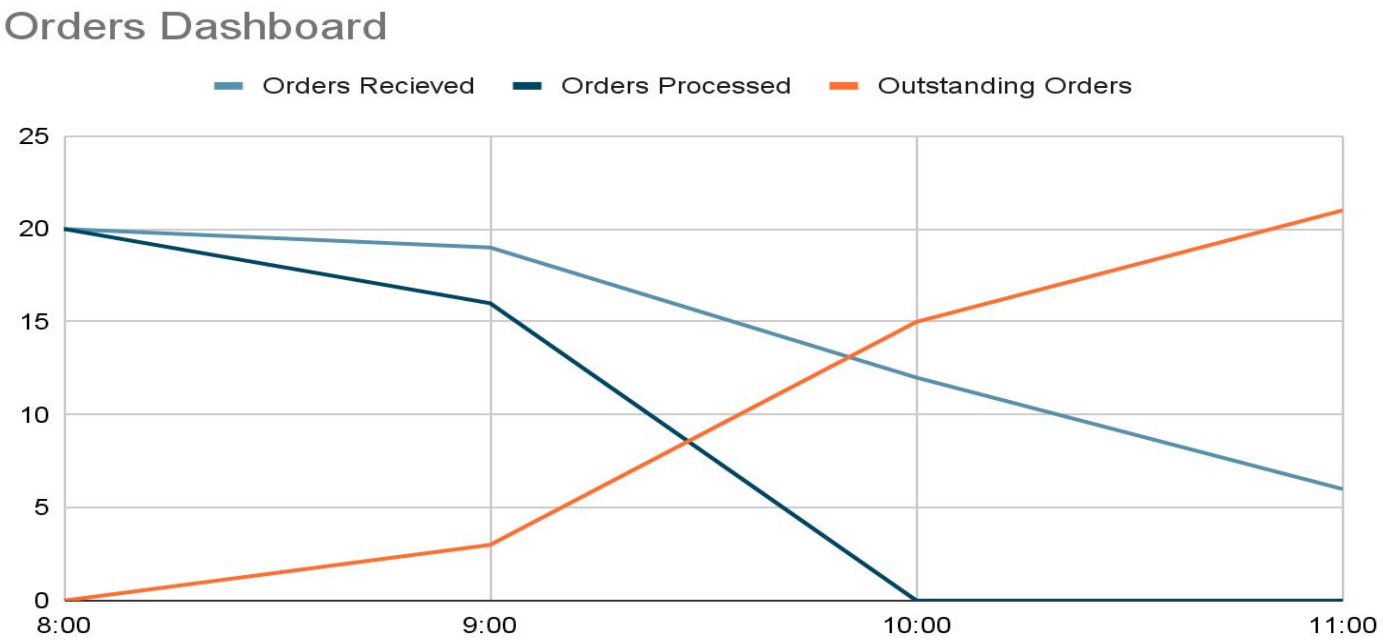
There are logs at /home/sre/logs/order_processing.log. If the server is not overloaded, this is a good place to check for errors. If you encounter any errors, send a message to the developers so that they can troubleshoot.

It is okay to restart this server during business hours. The Order Processor will pick up where it left off after a restart.

On-Call Shift -- Alert 1

Order Processing Issues, cont

Log/Monitoring Details




```
Order Processed
Processing Order 12
Order Processed
Processing Order 13
Order Processed
Processing Order 14
Order Processed
Processing Order 15
Order Processed
Processing Order 16
Order Processed
Processing Order 17
Order Processed
Processing Order 18
Order Processed
Processing Order 19
Order Processed
Processing Order 20
Order Processed
All Orders processed.

Startup...
Starting to process orders.
Processing Order 1
Order Processed
Processing Order 2
Order Processed
Processing Order 3
Order Processed
Processing Order 4
Error Processing Order. Error #12
Error Processing Order. Error #12
Error Processing Order. Error #12
Error Processing Order. Error #12
Error Processing Order. Error #12
Error Processing Order. Error #12
```

On-Call Shift -- Alert 2

Low Storage Alert

Summary

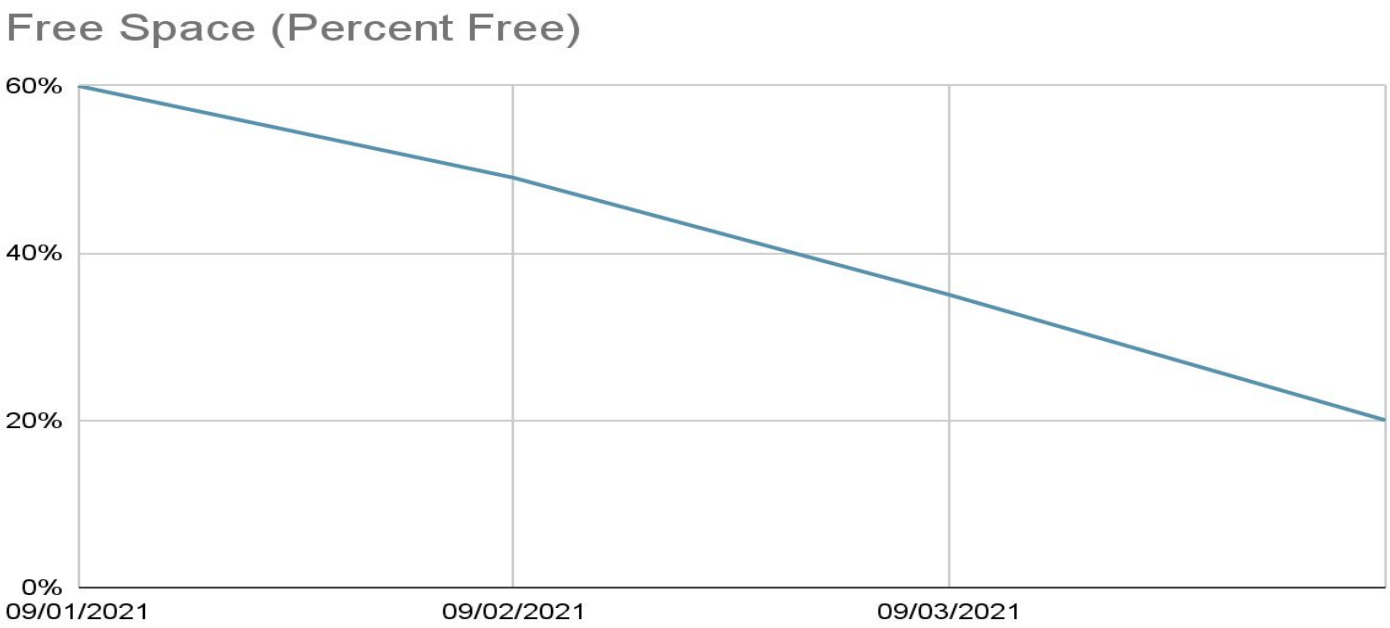
You receive an alert that the storage is running out on the mount where application logs are being written to. After consulting the SOP, you reach out to the team responsible for the server. They respond that Steve is normally in charge of handling the logs. Every morning he would run the commands listed in the run book, but he has been out sick for a week. The other members of the team forgot that it needed to be done, so the mount filled up.

SOP

Low Storage
Depending on the specific alert take the following action:

/home/sre/logs/app.log -- If this mount is low on storage, reach out to Compliance. They will know what logs can be cleared out or will request additional storage.

Log/Monitoring Details



On-Call Shift -- Alert 3

DNS Troubles

Summary

The networking team recently added a secondary backup DNS server to increase reliability since the one they are using now tends to go down frequently. Your team has several checks in place monitoring the DNS servers to make sure they are up at all times.

SOP

DNS Server Not Answering Requests

If you receive this alert, you should check to see if DNS1 or DNS2 is the current server answering requests. After determining which is the active server, check to see if the server is reachable. If the server is not reachable, immediately initiate the failover procedure to prevent any further network disruptions. If the server is reachable, check the logs to determine what the error is. If the active server cannot be brought back online within 5 mins, initiate the failover procedure. Either way, engage the Networking team to bring the standby server back online.

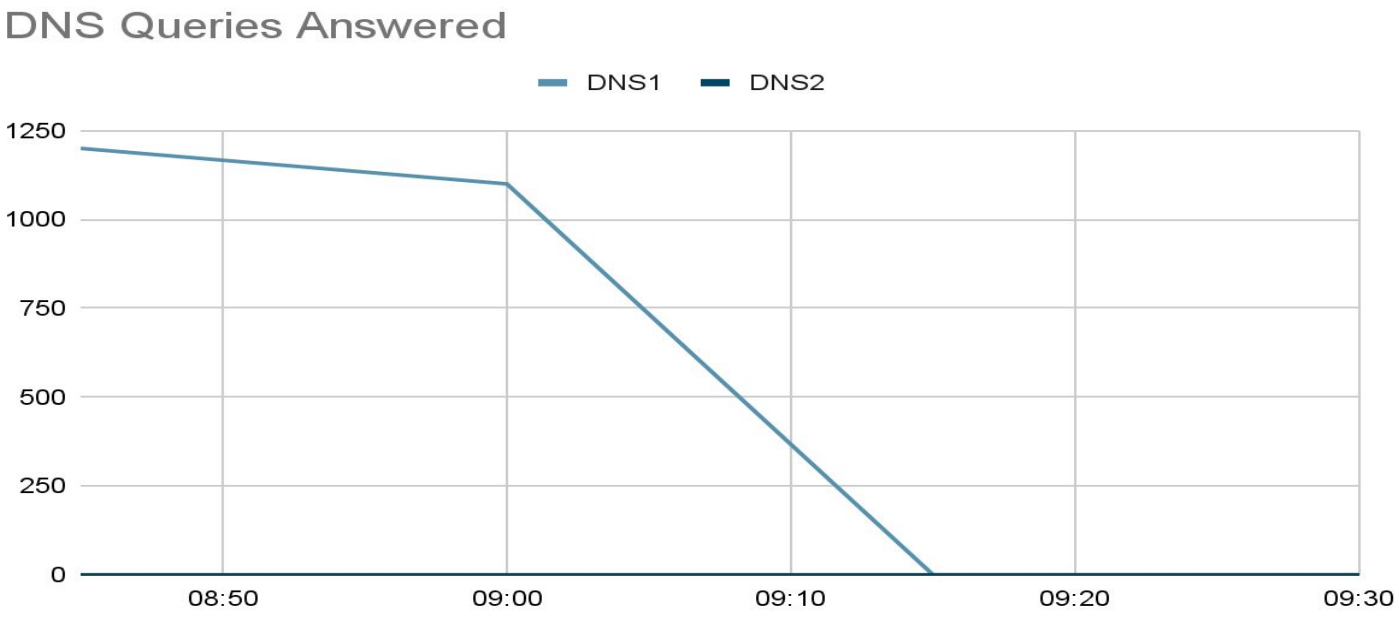
Failover Procedure

1. Determine the active server with the dnsTool.
    a. `dnsTool -q active_server`
2. If the active server is reachable you can initiate the shutdown process. If this command fails, make sure the dns process is shutdown on the server before continuing
    a. `dnsTool -a shutdown -s dns1`
3. Start the failover.
    a. If shutdown was successful:
    `dnsTool -a failover -s dns2`
    b. If shutdown was not successful, include the force flag,
    `dnsTool -a failover -s dns -f`

On-Call Shift -- Alert 3

DNS Troubles, cont

Log/Monitoring Details



Networking Server Status Page	
Server	Status
DNS1	UP
DNS2	UP

On-Call Shift -- Alert 3

DNS Troubles, cont

Log/Monitoring Details

```
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Answering Query.
Unexpected Error encountered.
Unexpected Error encountered.
Unexpected Error encountered.
Unexpected Error encountered.
Unexpected Error encountered.
Unexpected Error encountered.
Unexpected Error encountered.
Unexpected Error encountered.
Unexpected Error encountered.
```

On-Call Shift -- Alert 4

Application Outage

Summary

You receive the dreaded Application Down alert. Not only do you receive an alert for the application being down, but Customer Support also sent out a page to get all hands on deck for a report of the application being down.

SOP

Application Down

If you receive this alert, you need to act immediately. First, verify the application is indeed unreachable. If the application is unreachable, check to make sure the hosts are up and the application processes are running. You must start escalation for this immediately after verification the app is unreachable. Contact the following POCs:

Customer Support -- Susan Vega
Networking -- Bob Sparrow
Ops -- Glen Hammer
Database Admin -- Karen House
Development Team – Gal Tree

Log/Monitoring Details

Main App Status	
Endpoint or Host	Status
exoticplant.plant	UNREACHABLE
planthost1.internal	UP
planthost2.internal	UP
exoticplant.plant.internal	UNREACHABLE

On-Call Shift -- Alert 4

Application Outage, cont

Log/Monitoring, cont.

```
Info: Processing Request 407
Warming: Timeout. Retrying 428
Info: Processing Request 439
Warming: Timeout. Retrying 447
Warming: Timeout. Retrying 941
Warming: Timeout. Retrying 168
Warming: Timeout. Retrying 205
Warming: Timeout. Retrying 278
Info: Processing Request 439
Info: Placing Order 492
Warming: Timeout. Retrying 814
Info: Placing Order 520
Warming: Timeout. Retrying 662
Info: Processing Request 776
Info: Processing Request 548
Info: Processing Request 559
Warming: Timeout. Retrying 905
Info: Placing Order 948
Info: Placing Order 340
Error: Var is 10 RETRYING
```

On-Call Shift -- Alert 4

Application Outage, cont

Log/Monitoring, cont.

09:15 Hey we have reports of an application outage and we can not reach the app either. **FROM: svega**
09:16 I have an alert for that too. I'm looking at things now, will start a communication channel to coordinate. Checking logs and app servers now. **FROM: YOU**

09:20 -- !svega !bsparrow !ghammer !khouse !gtree we have an application outage **FROM: YOU**.
0930 -- Everything looks good from the network **FROM: sparrow**
0932 -- I can access the DB and it is reporting back normal **FROM: khouse**
0935 -- Everything here looks normal. FROM: ghammer
0937 -- We are still reviewing logs and seeing if we can reproduce on our end FROM: gtree
0938 -- We should try restarting the app, Maybe that will help FROM: ghammer
0940 -- Maybe that will help. FROM: svega
0943 -- Okay I will try. Bringing down. FROM: YOU
0945 -- App is down. Bring back up. FROM: YOU
0947 -- App is starting. FROM: YOU
0952 -- Main app is back up. FROM: hammer
0955 -- App is still not respond. FROM: svega
0956 -- I'm sending you some new logs !gtree these look off FROM: hammer
1005 -- !sre !ghammer when was the last deploy? What were the details? This looks like a qa build. FROM: gtree
1007 -- I did a deploy with one of the devs to qa to do some testing. Let me check. FROM: ghammer
1010 -- I think there was a mixup when doing the deployment. The wrong scripts was used and that build was deployed to prod. FROM hammer
1011 -- Were there any migrations for that !ghammer FROM: khouse
1012 -- No, just code changes. FROM: hammer
1013 -- Thats good. We should be able to just revert back then. !svega
1015 -- Let me take down the app and redeploy it. FROM: YOU
1017 -- App is down. Bring back up. FROM: YOU
1023 -- App is starting. FROM: YOU
1026 -- Main app is back up. FROM: hammer
1030 -- Everything looks like it is responding now. FROM: svega

On-Call Summary Log

September 4, 2021/11:05am -- Order Processing Issues Alert

Troubleshooting

- Checked the dashboard to see if the Order Processor is overloaded with orders
- Checked logs at /home/sre/logs/order_processing.log and found errors to process orders

Resolution

- Sent a message to the developers so that they can troubleshoot the errors to process orders
- Restarted this server and the Order Processor picked up where it left off after the restart

September 4, 2021/2:58pm -- Low Storage Alert

Troubleshooting

- Checked the percentage of free storage and it was 20%

Resolution

- Sent a message to Compliance and asked whether the logs can be cleared out or addition storage is needed

On-Call Summary Log

September 4, 2021/5:14pm -- DNS Troubles

Troubleshooting

- Checked whether DNS1 or DNS2 is the current server answering requests and found that DNS1 was answer queries
- Checked whether DNS1 and DNS2 are reachable and they are reachable
- Checked the logs and found the “unexpected error encountered” error

Resolution

- Shut down DNS1 and engaged the Networking team to bring DNS2 back online
- Started the failover to DNS2

September 4, 2021/9:15pm -- Application Outage

Troubleshooting

- Verified that the application was unreachable
- Checked the hosts, which were up
- Checked the application processes, which were running
- Contact the following POCs, Susan Vega, Bob Sparrow, Glen Hammer, Karen House and Gal Tree. Bob verified that network was good. Karen House verified database was normal. Gal recommended restarting the app, after which the app did not respond. Gal and Glen identified that the dev version of the app was deployed to production.

Resolution

- Took down the dev version of the app from the production env
- Deployed the prod version of the app to the production env

Post-Mortem

Application Outage -- Sep 4, 2021/10:30pm

Stakeholders

Customer Support -- Susan Vega
Networking -- Bob Sparrow
Ops -- Glen Hammer
Database Admin -- Karen House
Development Team – Gal Tree

Incident Timeline

The outage was reported by Susan at 9:15pm. The application was brought back at 10:26pm. Susan verified that everything was working at 10:30pm.

Impact

The outage affected the order processing system preventing orders from being processed. This led to customers having delayed orders, as well as having to pull additional business resources in to process orders manually. This led to a loss of revenue for the business.

Resolution

Gal and Glen identified that the dev version of the app was deployed to production. The dev version of the app was taken down from the production env. The prod version of the app was deployed to the production env.

Action Plan

Implement safeguards to prevent applications from being deployed to the wrong environments.

Scenario 3

Toil Reduction

Toil Reduction Plan

Summary

Now that you have spent some time on your own as an SRE, you now have to round out your week by handling some of the toil you encountered. Looking through the on-call summary, post-mortem, as-built design doc, and your experience, you decided that there are several ways to reduce toil. You start by listing out 5 of the major items for this week. For each one, you analyze the impact on the business and what you gain by automating the task. After that, you will need to implement three of these items in pseudocode to help your team move forward.

Current Toil Items

1. The established release process is a manual affair generally done by one of the operations team members. Because it is manually done, the process is inconsistent and susceptible to mistakes. It is more costly due to the time spent on these tasks repetitively and the resources utilized from the operations team members.
2. The OPs engineer would add new resources as needed during the release window. This has led to downtime in the past as the app became overloaded and could not serve requests anymore.
3. Configuration management is a manual process. In the past this has caused issues. For example, steps have been forgotten or not all the scripts were executed.
4. DNS failover is a manual process.
5. One team member is in charge of handling the logs and run the commands listed in the run book every morning. If this team member is out of the office, the log storage will be filled up.

The established release process is a manual affair generally done by one of the operations team members. The OPs team generally will download the latest code, shut down the app, run the database migrations, change or add any needed configurations and then start the app back up. In the past this has caused issues as steps have been forgotten, not all the scripts were executed, the app was not restarted properly, among other issues. During the release window, the OPs engineer would also add new resources as needed. This has led to downtime in the past as the app became overloaded and could not serve requests anymore.

Automation Implementation

The release process can automated by using a CICD pipeline. For example, GitHub/BitBucket and CircleCI/Jenkins/ArgoCD can be used to implement the CICD pipeline. Below is an example of the declarative file if ArgoCD is used for continuous deployment.

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
 name: mainApp
 namespace: prod
spec:
 destination:
   namespace: default
   server: https://kubernetes.default.svc
 project: default
 source:
   helm:
     valueFiles:
     - values-prod.yaml
   path: helm
   repoURL: https://github.com/exoticplant/app
   targetRevision: HEAD
```

Automation Implementation

Resource/Infrastructure deployment should be automated using terraform and performed prior to the release window. Below is an example of the terraform file if azure virtual machines are deployed using terraform.

```
provider "azurerm" {
 features {}
}

resource "azurerm_linux_virtual_machine" "main" {
 count                           = var.instance_count
 name                            = "vm-${count.index}"
 resource_group_name             = var.resource_group
 location                        = var.location
 size                            = "Standard_D2s_v3"
 source_image_id                 = var.image_resource_id
 availability_set_id             = azurerm_availability_set.avset.id
 admin_username                  = "vmadmin"
 admin_password                  = "p@ssword1234"
 disable_password_authentication = false
 network_interface_ids           =
[element(azurerm_network_interface.main.*.id, count.index)]


 tags = {
   DevOps = "Deploy a Web Server in Azure"
 }
}
```

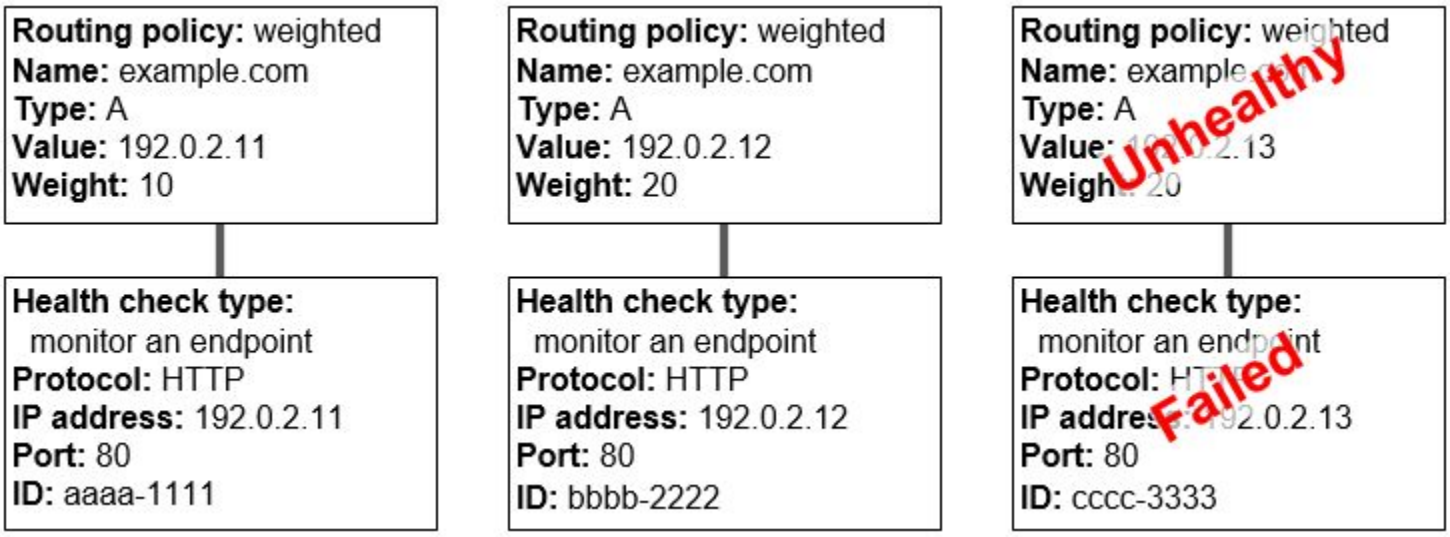
Automation Implementation

Configuration management can be automated by using tools such as Ansible, Chef and Puppet. Below is an example of of the declarative file if Ansible is used to configure the EC2 instances.

```
- name: Configuring EC2 instances
 hosts: ubuntu
 user: ubuntu
 become: true
 become_method: sudo
 become_user: root
 gather_facts: false
 vars:
   - ansible_python_interpreter: /usr/bin/python3
   - ansible_host_key_checking: false
   - ansible_stdout_callback: yaml
 environment:
   - TYPEORM_CONNECTION: "{{ lookup('env', 'TYPEORM_CONNECTION')}}"
   - TYPEORM_ENTITIES: "{{ lookup('env', 'TYPEORM_ENTITIES')}}"
   - TYPEORM_HOST: "{{ lookup('env', 'TYPEORM_HOST')}}"
   - TYPEORM_PORT: 5432
   - TYPEORM_USERNAME: "{{ lookup('env', 'TYPEORM_USERNAME')}}"
   - TYPEORM_PASSWORD: "{{ lookup('env', 'TYPEORM_PASSWORD')}}"
   - TYPEORM_DATABASE: "{{ lookup('env', 'TYPEORM_DATABASE')}}"
   - TYPEORM_MIGRATIONS: "{{ lookup('env', 'TYPEORM_MIGRATIONS')}}"
   - TYPEORM_MIGRATIONS_DIR: "{{ lookup('env', 'TYPEORM_MIGRATIONS_DIR')}}"
 roles:
   - configure-server
   - configure-prometheus-node-exporter
```

Automation Implementation

Automate DNS failover process using tools such as Amazon Route 53. For example, Amazon Route 53 can be configured to check the health of the servers and respond to DNS queries using only the healthy servers.



Automation Implementation

Implement automated log rotation using tools such as logrotate. Logrotate helps to administer logs, compress them, remove them or even email them after a certain time period. Below is an example configuration file.

```
/var/lib/docker/containers/*/*.log #The path where logrotate will monitor all the log files {

    rotate 5 #Store maximum 5 files of old logs when rotation hits
    Copytruncate #Truncates the original log file to zero size in place after creating a copy
Missingok #If the log file is missing, do not generate an error, and move on the next file
    Notifempty #Do not rotate the log if it is empty
    Compress #Old version of logs are compressed with gzip format
    maxsize 200M #Rotate the log file if it exceeds 200 mb in size, regardless of the rotation time unit
    Daily #Rotation should happen daily

}
```