# A. Introduction of Solution

**Purpose**

The solution purpose is to provide a flexible and scalable solution for the latest installment of SparkIT-Game Company’s newest release flagship game Capack Galactic 2 Treasure Planet.

**Goals and Objectives**

The goal and objective of the solution are to meet scaling needs of SpartIT-Game Company ahead of the upcoming PAX West conference to help ensure that there are no issues during the conference as this game will be part of a live game mode media stream with professional CG players. This solution includes scaling the clusters to reliably handle up to eight million concurrent users while also utilizing dynamic scaling that should scale back the clusters in non-peak times to make them more cost effective for SparkIT-Game Company.

SparkIT-Game Company would like to utilize modern tools and frameworks currently available on the market to best leverage the latest capabilities as well as encourage top talent to join their team. To meet this goal, the solution with leverage the AWS EKS hosted Kubernetes Service and follow Open Container Initiative specifications. The solution will leverage Chef to deploy the CRI-O container runtime purpose built for Kubernetes as the contain host for the platform. To maintain redundancy the CRI-O container host running under the EKS Kubernetes control plane will host containerized instances of all required applications and resources in each cluster.

**Scope**

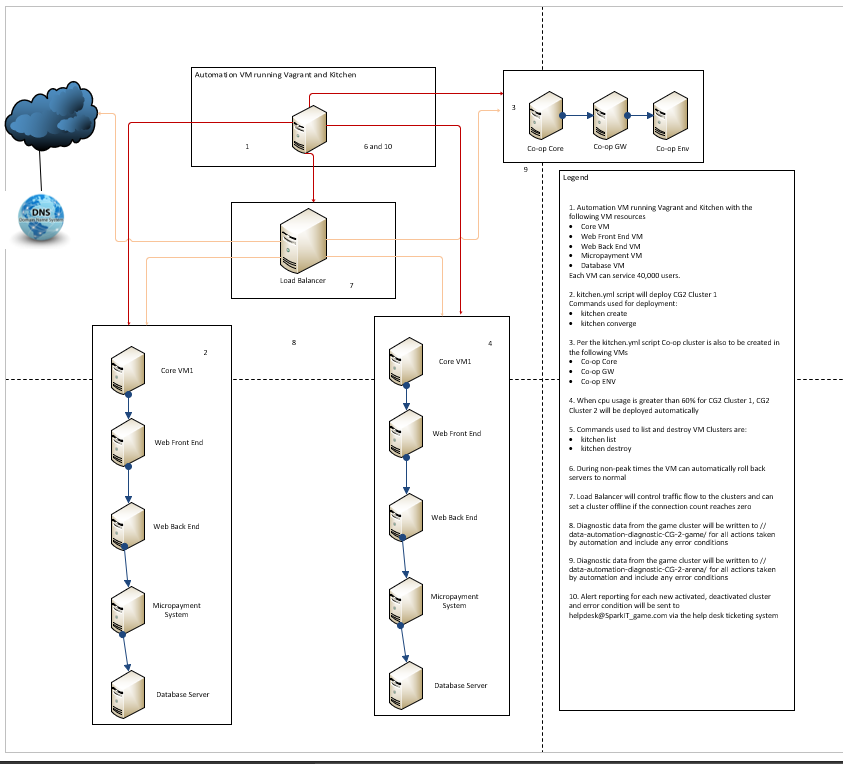
A CloudWatch metric will be used to track the RequestCountPerTarget with a threshold of 1000 requests that individual application targets can handle. Each cluster has 40 targets for a total 40,000 concurrent users per cluster. For proactivity, a new cluster will being to spin up when the backend requests are 30,000 for a cluster currently in service. Since cooperative play cannot handle as many requests micropayment, core, and backend clusters they will begin to scale at 20,000 requests instead of 30,000. Then the total numbers of requests are greater than 40,000 more than the total possible requests for backend, core micropayment clusters, and 20,000 for cooperative play clusters, the number fo clusters will then scale back to save money. When new clusters are created or destroyed an AWS SNS notification will be sent to [helpdesk@SparkIT\_game.com](mailto:helpdesk@SparkIT_game.com).

AWS EC2 Autoscaling Group will also remove clusters when CloudWatch logs indicate that there are high numbers of errors occurring in a particular cluster. Alters about elevated errors will also trigger and AWS SNS notification that will be sent to [helpdesk@SparktIT\_game.com](mailto:helpdesk@SparktIT_game.com).

**Functionality**

To deploy the solution there are several tools and platforms that will utilized for this solution. First will be the EKS control plane, EC2 load balancers, CloudWatch metrics and alarms, EC2 AutoScaling Group and DNS records will be deployed with Terraform. When new servers and spun up in the AutoScaling Group, Chef cookbooks will run to install the CRI-O platform for containerized hosting and associate the hosts with the Kubernetes control plane. Finally, the platform should add the additional clusters and scale out as needed in peak times and scale in when utilization is lower in non-peak times to save resources and save money to be const effective. The helpdesk will be notified with scaling events occur and/or when errors occur on the platform.

# B. Visual Representation



# C. Automation Script

1. Copy your code from part C here :

---

driver:

name: vagrant

network:

- ["private\_network", { type: "dhcp" }]

# driver:

# name: ec2

# aws\_ssh\_key\_id: MyAWSKeyPair

# region: us-east-2

# availability\_zone: b

# subnet\_id: subnet-f46f388e

# instance\_type: t2.micro

# image\_id: ami-08ee2516c7709ea48

# security\_group\_ids: ["sg-ccd501ae"]

# retryable\_tries: 120

provisioner:

name: chef\_zero

verifier:

name: inspec

platforms:

- name: centos-7

transport:

ssh\_key: sshfiles/MyAWSKeyPair.pem

username: centos

connection\_timeout: 10

connection\_retries: 5

suites:

- name: base\_core

provisioner:

named\_run\_list: base\_core

verifier:

inspec\_tests:

- test/integration/base\_core

attributes:

tags:

Name: 'Base-Game-Core'

Cluster: 'Base'

- name: base\_web\_front\_end

provisioner:

named\_run\_list: web\_front\_end

verifier:

inspec\_tests:

- test/integration/web\_front\_end

attributes:

tags:

Name: 'Base-Game-Frontend'

Cluster: 'Base'

- name: base\_web\_back\_end

provisioner:

named\_run\_list: web\_back\_end

verifier:

inspec\_tests:

- test/integration/web\_back\_end

attributes:

tags:

Name: 'Base-Game-Backend'

Cluster: 'Base'

- name: base\_database

provisioner:

named\_run\_list: database

verifier:

inspec\_tests:

- test/integration/database

attributes:

tags:

Name: 'Base-Game-Database'

Cluster: 'Base'

- name: base\_micro\_payment\_server

provisioner:

named\_run\_list: micro\_payment\_server

verifier:

inspec\_tests:

- test/integration/micro\_payment\_server

attributes:

tags:

Name: 'Game-Micropayment-Server'

Cluster: 'Base'

- name: coop\_core

provisioner:

named\_run\_list: coop\_core

verifier:

inspec\_tests:

- test/integration/coop\_core

attributes:

tags:

Name: 'Coop-Core'

Cluster: 'Coop'

- name: coop\_gateway

provisioner:

named\_run\_list: coop\_gateway

verifier:

inspec\_tests:

- test/integration/coop\_gateway

attributes:

tags:

Name: 'Coop-Gateway'

Cluster: 'Coop'

- name: coop\_environment

provisioner:

named\_run\_list: coop\_environment

verifier:

inspec\_tests:

- test/integration/coop\_environment

attributes:

tags:

Name: 'Coop-Env'

Cluster: 'Coop'

1. Screenshot showing that the automation script executes without errors (from part D):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Screenshot 1 | Screenshot 2 | Screenshot 3 | Screenshot 4 | Screenshot 5 | Screenshot 6 |

# D. Diagnostic Report

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Description** | **Optimal Range** | **Data and Results** | **Script Used to Extract Data** | **Screenshot of Result of Script** |
| Time to scale from 1 cluster to 200 clusters  (8 million users expected at peak after PAX West) based on 40K users per cluster (subject to change based on load testing) | 15–30 minutes for each cluster | Build time for single cluster = 9m29.75s  Build for 200 clusters =  9.20 \* 200 = 1860 minutes (rounded) or 31 Hours/1.29 Days | Kitchen create  Kitchen list (to verify) | Kitchen create    Kitchen list |
| Time to register a cluster and then quench connections to the load balancer, taking the cluster off-line (start-up, operation, shutdown) | 1 minute per connection quench, start of cluster launch, and part of time to scale cluster, can be tracked separately as a quench | 1m26.54s | kitchen destroy |  |
| Peak load averages per system at 20K, 30K, and 40K users per game cluster | 60% of CPU triggers new cluster launch; if reaching core load at 20K users, launch new cluster on 60% CPU loads | Load average = 0.00. 0.01, 0.05 | kitchen exec base-core-centos-7 -c ‘top’ |  |
| Write times to the diagnostic data drive | <30 milliseconds | Random Read - 1.21 MB/s  Sequential Read - 143.23 MB/s  Sequential Write - 132.98 MB/s  Avg. Read with Seq. Writes – 2.762 ms  Avg. Read with Random Writes – 7.178 ms | Winsat disk -drive g |  |
| Pull time from the game instances (1 core, 1 web front end, 1 web back end, 1 database, and 1 micropayment server) and initialization time | Part of cluster launch 15–30 minutes | 9m22.10s | Kitchen converge |  |
| Average messaging service (queue) time | <1 minute in queue |  |  | N/A |
| Average latency for the micropayment server | <30 milliseconds | rtt min/avg/max/mdev = 32.811/36.028/43.268/4.264 ms | Ping google.com |  |
| Average latency of each cluster | <30 milliseconds | rtt min/avg/max/mdev = 0.011/0.023/0.030/0.005 ms | Ping localhost |  |
| Network data in and out for each cluster | <1 second | rtt min/avg/max/mdev = 32.369/34.619/38.971/2.202 ms | kitchen exec base-core-centos-7 -c 'ping -c 6 google.com' |  |
| Overall CPU utilization of the environment for each cluster | Not >60% | load average: 0.21, 0.06, 0.06 | kitchen exec coop-core-centos-7 -c 'top' |  |
| Diagnostic data able to be written by the automation to the correct cloud bucket storage space | Show read/write times <1 second |  |  | N/A |
| Cooperative play cluster latency | <30 milliseconds | rtt min/avg/max/mdev = 0.012/0.020/0.023/0.005 ms | kitchen exec coop-core-centos-7 -c 'ping -c 6 localhost' |  |
| Cooperative play latency between gateway/matching and core | <30 milliseconds | rtt min/avg/max/mdev = 0.013/0.021/0.024/0.006 ms | kitchen exec coop-gateway-centos-7 -c 'ping -c 6 localhost' |  |
| Cooperative play latency between gateway/matching and environment | <30 milliseconds | rtt min/avg/max/mdev = 32.007/35.629/39.482/2.607 ms | kitchen exec coop-gateway-centos-7 -c 'ping -c 6 google.com' |  |
| Pull time from the cooperative play instances and initialization time | 15–30 minutes for each cluster | 1m52.01s | Kitchen create  Kitchen list (to verify) | Kitchen create    Kitchen list |

# E. Web Sources

About the Open Container Initiative. (n.d.). Retrieved August 01, 2020, from https://opencontainers.org/about/overview

Chef Supermarket. (n.d.). Retrieved August 01, 2020, from https://supermarket.chef.io/cookbooks-directory

Container runtimes. (2020, June 23). Retrieved August 01, 2020, from https://kubernetes.io/docs/setup/production-environment/container-runtimes

Hansen, N. (1973). Amazon Elastic Kubernetes Service. Retrieved August 01, 2020, from https://aws.amazon.com/eks/

Kitchen.yml. (n.d.). Retrieved August 01, 2020, from https://docs.chef.io/config\_yml\_kitchen.html

KitchenCI. (n.d.). Retrieved August 01, 2020, from https://kitchen.ci/docs/getting-started/introduction/

Learn Chef. (n.d.). Retrieved August 01, 2020, from https://learn.chef.io/modules/local-development/ubuntu

Sous-Chefs. (n.d.). Sous-chefs/apache2. Retrieved August 01, 2020, from https://github.com/sous-chefs/apache2

UAlaska-IT. (n.d.). UAlaska-IT/sqlite\_install. Retrieved August 01, 2020, from https://github.com/ualaska-it/sqlite\_install