EIPAAS-5551 Implementation plan(Kubernetes and RDS Sensu Alerts)

Overview

Defining alerts in SM Prometheus and updating sensu to check for Prometheus alerts and raise individual tickets for each one and for RDS instances.

Service Checks

Make sure on consul all services are up and running.

Make sure sensu is working as expected and creating alerts on to service desk.

check the SM Prometheus status and the existing rules.

Assumptions

SM Prometheus is already provisioned on the metric store node and sensu monitoring.

Hot Fixes

Ensure the following commits are merged into the project branches

Repo	Commit
kubernetes-config	e8f21bd121b f8ba42719e9
profile	88216a7a091

Pre-Implementation Plan

- SSH into the metric store server and take the backup of prometheus_rules.ctmpl and prometheus.ctmpl.

```
cp /etc/consul-template/prometheus_rules.ctmpl /etc/consul-template/prometheus_rules.ctmpl-bkp
cp /etc/consul-template/prometheus.ctmpl /etc/consul-template/prometheus.ctmpl-bkp
```

- SSH into the monitoring server and take the backup of sensu_container_checks.ctmpl, jira_handler.rb, and nkaas.json.

```
cp /etc/consul-template/sensu_container_checks.ctmpl /etc/consul-template/sensu_container_checks.ctmpl-
bkp
cp /opt/jira_handler.rb /opt/jira_handler.rb-bkp
cp /etc/consul.d/seed_kv/sensu/nkaas.json /etc/consul.d/seed_kv/sensu/nkaas.json-bkp
```

Implementation Plan

1. Update the Bitbucket code to pick new Prometheus rules and sensu changes.
2. Update the Prometheus rules.
    a. SSH to the Metric store.
    b. Update the prometheus target template file to create new target(cloud exporter) on Prometheus.
    c. /etc/consul-template/prometheus.ctmpl(add the below code into the native kubernetes section)

prometheus.ctmpl

```
# cloudwatch-exporter
- job_name: '{{ $dc.Value }}-cloudwatch-exporter'
  tls_config:
    ca_file: /client-bundles/{{ $dc.Value }}/ca.crt
  scheme: https
  bearer_token: {{ with secret ( print "internal/management" ( print "/sm_metric_store/prometheus
/sa/" $env "-kubernetes" ) ) }}{{ .Data.data.token }}{{ end }}
  static_configs:
    - targets: ['{{ key (printf "%s@%s" "kubernetes/api_server_url" $dc.Value) }}:6443']
      labels:
        __metrics_path__: /api/v1/namespaces/xpaas-prometheus-cloudwatch-exporter/services
/prometheus-cloudwatch-exporter:9106/proxy/metrics
        environment: {{ $dc.Value }}
```

d. Update the below file to create new rules on Prometheus.
e. /etc/consul-template/prometheus_rules.ctmpl

prometheus_rules.ctmpl

```
groups:
  - name: grafana
    rules:
    - record: super_info
      expr: label_replace((kube_pod_info{created_by_kind="ReplicaSet"}), "replicaset", "$1",
"created_by_name", "(.*)") * on (replicaset,environment,namespace) group_left (owner_name)
(kube_replicaset_owner) OR label_replace((kube_pod_info{created_by_kind!="ReplicaSet"}),
"owner_name", "$1", "created_by_name", "(.*)")
    - record: super_container_cpu
      expr: sum(rate(container_cpu_usage_seconds_total{container!="", container!="POD"}[5m]) * on
(pod,environment,namespace) group_left(owner_name, host_ip) super_info{owner_name!="<none>"}) by
(owner_name, host_ip, pod, container, namespace)
    - record: super_container_memory
      expr: avg(container_memory_usage_bytes{container!="", container!="POD"} * on (pod,
environment,namespace) group_left(owner_name, host_ip) super_info{owner_name!="<none>"}) by
(owner_name, host_ip, pod, container, namespace)
    - record: super_network_in
      expr: sum (rate(container_network_receive_bytes_total[5m]) * on(pod,environment,namespace)
group_left(owner_name, host_ip) super_info{owner_name!="<none>"}) by (owner_name, host_ip, pod,
container, namespace)
    - record: super_network_out
      expr: sum(rate(container_network_transmit_bytes_total[5m]) * on (pod,environment,namespace)
group_left(owner_name, host_ip) super_info{owner_name!="<none>"}) by (owner_name, host_ip, pod,
container, namespace)
    - record: super_container_fs_read
      expr: sum(rate(container_fs_reads_total{container!="", container!="POD"}[5m]) * on (pod,
environment,namespace) group_left(owner_name, host_ip) super_info{owner_name!="<none>"}) by
(owner_name, host_ip, pod, container, namespace)
    - record: super_container_fs_write
      expr: sum(rate(container_fs_writes_total{container!="", container!="POD"}[5m]) * on (pod,
environment,namespace) group_left(owner_name, host_ip) super_info{owner_name!="<none>"}) by
(owner_name, host_ip, pod, container, namespace)
  - name: general.rules
    rules:
    - alert: TargetDown
      annotations:
        description: '{{`{{ printf "%.4g" $value }}`}}% of the {{`{{ $labels.job }}`}}/{{`{{
$labels.service }}`}} targets in {{`{{ $labels.namespace }}`}} namespace are down.'
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-targetdown
        summary: One or more targets are unreachable.
      expr: 100 * (count(up == 0) BY (environment, clusterName, job, namespace, service) /
        count(up) BY (environment, clusterName, job, namespace, service)) > 10
      for: 10m
      labels:
```

```
        severity: warning
    - alert: Watchdog
      annotations:
        description: |
          This is an alert meant to ensure that the entire alerting pipeline is functional.
          This alert is always firing, therefore it should always be firing in Alertmanager
          and always fire against a receiver. There are integrations with various notification
          mechanisms that send a notification when this alert is not firing. For example the
          "DeadMansSnitch" integration in PagerDuty.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-watchdog
        summary: An alert that should always be firing to certify that Alertmanager
          is working properly.
      expr: vector(1)
      labels:
        severity: none
  - name: k8s.rules
    rules:
    - expr: |-
        sum by (environment, clusterName, namespace, pod, container) (
          irate(container_cpu_usage_seconds_total{image!=""}[5m])
        ) * on (environment, clusterName, namespace, pod) group_left(node) topk by (environment,
clusterName, namespace, pod) (
          1, max by(environment, clusterName, namespace, pod, node) (kube_pod_info{node!=""})
        )
      record: node_namespace_pod_container:container_cpu_usage_seconds_total:sum_irate
    - expr: |-
        container_memory_working_set_bytes{
          image!=""}
        * on (environment, clusterName, namespace, pod) group_left(node) topk by(environment,
clusterName, namespace, pod) (1,
          max by(environment, clusterName, namespace, pod, node) (kube_pod_info{node!=""})
        )
      record: node_namespace_pod_container:container_memory_working_set_bytes
    - expr: |-
        container_memory_rss{image!=""}
        * on (namespace, pod) group_left(node) topk by(namespace, pod) (1,
          max by(namespace, pod, node) (kube_pod_info{node!=""})
        )
      record: node_namespace_pod_container:container_memory_rss
    - expr: |-
        container_memory_cache{image!=""}
        * on (namespace, pod) group_left(node) topk by(namespace, pod) (1,
          max by(namespace, pod, node) (kube_pod_info{node!=""})
        )
      record: node_namespace_pod_container:container_memory_cache
    - expr: |-
        container_memory_swap{image!=""}
        * on (namespace, pod) group_left(node) topk by(namespace, pod) (1,
          max by(namespace, pod, node) (kube_pod_info{node!=""})
        )
      record: node_namespace_pod_container:container_memory_swap
    - expr: |-
        kube_pod_container_resource_requests{resource="memory"}  * on (namespace, pod, clusterName)
        group_left() max by (namespace, pod, clusterName) (
          (kube_pod_status_phase{phase=~"Pending|Running"} == 1)
        )
      record: cluster:namespace:pod_memory:active:kube_pod_container_resource_requests
    - expr: |-
        sum by (namespace, clusterName, environment) (
            sum by (namespace, pod, clusterName, environment) (
                max by (namespace, pod, container, clusterName, environment) (
                  kube_pod_container_resource_requests{resource="memory"}
                ) * on(namespace, pod, clusterName, environment) group_left() max by (namespace,
pod, clusterName, environment) (
                  kube_pod_status_phase{phase=~"Pending|Running"} == 1
                )
            )
        )
      record: namespace_memory:kube_pod_container_resource_requests:sum
    - expr: |-
```

```
        kube_pod_container_resource_requests{resource="cpu"}  * on (namespace, pod, clusterName,
environment)
        group_left() max by (namespace, pod, clusterName) (
          (kube_pod_status_phase{phase=~"Pending|Running"} == 1)
        )
      record: cluster:namespace:pod_cpu:active:kube_pod_container_resource_requests
    - expr: |-
        sum by (namespace, clusterName, environment) (
            sum by (namespace, pod, clusterName, environment) (
                max by (namespace, pod, container, clusterName, environment) (
                  kube_pod_container_resource_requests{resource="cpu"}
                ) * on(namespace, pod, clusterName, environment) group_left() max by (namespace,
pod, clusterName, environment) (
                  kube_pod_status_phase{phase=~"Pending|Running"} == 1
                )
            )
        )
      record: namespace_cpu:kube_pod_container_resource_requests:sum
    - expr: |-
        kube_pod_container_resource_limits{resource="memory"}  * on (namespace, pod, clusterName)
        group_left() max by (namespace, pod, clusterName) (
          (kube_pod_status_phase{phase=~"Pending|Running"} == 1)
        )
      record: cluster:namespace:pod_memory:active:kube_pod_container_resource_limits
    - expr: |-
        sum by (namespace, clusterName, environment) (
            sum by (namespace, pod, clusterName, environment) (
                max by (namespace, pod, container, clusterName, environment) (
                  kube_pod_container_resource_limits{resource="memory"}
                ) * on(namespace, pod, clusterName, environment) group_left() max by (namespace,
pod, clusterName, environment) (
                  kube_pod_status_phase{phase=~"Pending|Running"} == 1
                )
            )
        )
      record: namespace_memory:kube_pod_container_resource_limits:sum
    - expr: |-
        kube_pod_container_resource_limits{resource="cpu"}  * on (namespace, pod, clusterName)
        group_left() max by (namespace, pod, clusterName) (
          (kube_pod_status_phase{phase=~"Pending|Running"} == 1)
          )
      record: cluster:namespace:pod_cpu:active:kube_pod_container_resource_limits
    - expr: |-
        sum by (namespace, clusterName, environment) (
            sum by (namespace, pod, clusterName, environment) (
                max by (namespace, pod, container, clusterName, environment) (
                  kube_pod_container_resource_limits{resource="cpu"}
                ) * on(namespace, pod, clusterName, environment) group_left() max by (namespace,
pod, clusterName, environment) (
                  kube_pod_status_phase{phase=~"Pending|Running"} == 1
                )
            )
        )
      record: namespace_cpu:kube_pod_container_resource_limits:sum
    - expr: |-
        max by (environment, clusterName, namespace, workload, pod) (
          label_replace(
            label_replace(
              kube_pod_owner{owner_kind="ReplicaSet"},
              "replicaset", "$1", "owner_name", "(.*)"
            ) * on(replicaset, namespace) group_left(owner_name) topk by(replicaset, namespace) (
              1, max by (replicaset, namespace, owner_name) (
                kube_replicaset_owner
              )
            ),
            "workload", "$1", "owner_name", "(.*)"
          )
        )
      labels:
        workload_type: deployment
      record: namespace_workload_pod:kube_pod_owner:relabel
```

```
    - expr: |-
        max by (environment, clusterName, namespace, workload, pod) (
          label_replace(
            kube_pod_owner{owner_kind="DaemonSet"},
            "workload", "$1", "owner_name", "(.*)"
          )
        )
      labels:
        workload_type: daemonset
      record: namespace_workload_pod:kube_pod_owner:relabel
    - expr: |-
        max by (environment, clusterName, namespace, workload, pod) (
          label_replace(
            kube_pod_owner{owner_kind="StatefulSet"},
            "workload", "$1", "owner_name", "(.*)"
          )
        )
      labels:
        workload_type: statefulset
      record: namespace_workload_pod:kube_pod_owner:relabel
  - name: kube-prometheus-general.rules
    rules:
    - expr: count without(instance, pod, node) (up == 1)
      record: count:up1
    - expr: count without(instance, pod, node) (up == 0)
      record: count:up0
  - name: kubernetes-apps
    rules:
    - alert: KubePodCrashLooping
      annotations:
        description: Pod {{`{{ $labels.namespace }}`}}/{{`{{ $labels.pod }}`}} ({{`{{ $labels.
container }}`}}) is restarting {{`{{ printf "%.2f" $value }}`}} times / 10 minutes.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubepodcrashlooping
        summary: Pod is crash looping.
      expr: |-
        increase(kube_pod_container_status_restarts_total{namespace=~".*"}[10m]) > 0
        and
        kube_pod_container_status_waiting{namespace=~".*"} == 1
      for: 15m
      labels:
        severity: warning
    - alert: KubePodNotReady
      annotations:
        description: Pod {{`{{ $labels.namespace }}`}}/{{`{{ $labels.pod }}`}} has been in a non-
ready
          state for longer than 15 minutes.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubepodnotready
        summary: Pod has been in a non-ready state for more than 15 minutes.
      expr: |-
        sum by (namespace, pod, clusterName, environment) (
          max by(namespace, pod, clusterName, environment) (
            kube_pod_status_phase{namespace=~".*", phase=~"Pending|Unknown"}
          ) * on(namespace, pod, clusterName, environment) group_left(owner_kind) topk by
(namespace, pod, clusterName, environment) (
            1, max by(namespace, pod, owner_kind, clusterName, environment) (kube_pod_owner
{owner_kind!="Job"})
          )
        ) > 0
      for: 15m
      labels:
        severity: warning
    - alert: KubeDeploymentGenerationMismatch
      annotations:
        description: Deployment generation for {{`{{ $labels.namespace }}`}}/{{`{{ $labels.
deployment }}`}} does not match, this indicates that the Deployment has failed but has not
          been rolled back.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubedeploymentgenerationmismatch
        summary: Deployment generation mismatch due to possible roll-back
```

```
      expr: |-
        kube_deployment_status_observed_generation{namespace=~".*"}
          !=
        kube_deployment_metadata_generation{namespace=~".*"}
      for: 15m
      labels:
        severity: warning
    - alert: KubeDeploymentReplicasMismatch
      annotations:
        description: Deployment {{`{{ $labels.namespace }}`}}/{{`{{ $labels.deployment }}`}} has
          not matched the expected number of replicas for longer than 15 minutes.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubedeploymentreplicasmismatch
        summary: Deployment has not matched the expected number of replicas.
      expr: |-
        (
          kube_deployment_spec_replicas{namespace=~".*"}
            >
          kube_deployment_status_replicas_available{namespace=~".*"}
        ) and (
          changes(kube_deployment_status_replicas_updated{namespace=~".*"}[10m])
            ==
          0
        )
      for: 15m
      labels:
        severity: warning
    - alert: KubeStatefulSetReplicasMismatch
      annotations:
        description: StatefulSet {{`{{ $labels.namespace }}`}}/{{`{{ $labels.statefulset }}`}} has
          not matched the expected number of replicas for longer than 15 minutes.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubestatefulsetreplicasmismatch
        summary: Deployment has not matched the expected number of replicas.
      expr: |-
        (
          kube_statefulset_status_replicas_ready{namespace=~".*"}
            !=
          kube_statefulset_status_replicas{namespace=~".*"}
        ) and (
          changes(kube_statefulset_status_replicas_updated{namespace=~".*"}[10m])
            ==
          0
        )
      for: 15m
      labels:
        severity: warning
    - alert: KubeStatefulSetGenerationMismatch
      annotations:
        description: StatefulSet generation for {{`{{ $labels.namespace }}`}}/{{`{{ $labels.
statefulset }}`}} does not match, this indicates that the StatefulSet has failed but has
          not been rolled back.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubestatefulsetgenerationmismatch
        summary: StatefulSet generation mismatch due to possible roll-back
      expr: |-
        kube_statefulset_status_observed_generation{namespace=~".*"}
          !=
        kube_statefulset_metadata_generation{namespace=~".*"}
      for: 15m
      labels:
        severity: warning
    - alert: KubeStatefulSetUpdateNotRolledOut
      annotations:
        description: StatefulSet {{`{{ $labels.namespace }}`}}/{{`{{ $labels.statefulset }}`}}
update
          has not been rolled out.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubestatefulsetupdatenotrolledout
        summary: StatefulSet update has not been rolled out.
      expr: |-
```

```
        (
          max without (revision) (
            kube_statefulset_status_current_revision{namespace=~".*"}
              unless
            kube_statefulset_status_update_revision{namespace=~".*"}
          )
            *
          (
            kube_statefulset_replicas{namespace=~".*"}
              !=
            kube_statefulset_status_replicas_updated{namespace=~".*"}
          )
        )  and (
          changes(kube_statefulset_status_replicas_updated{namespace=~".*"}[5m])
            ==
          0
        )
      for: 15m
      labels:
        severity: warning
    - alert: KubeDaemonSetRolloutStuck
      annotations:
        description: DaemonSet {{`{{ $labels.namespace }}`}}/{{`{{ $labels.daemonset }}`}} has not
          finished or progressed for at least 15 minutes.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubedaemonsetrolloutstuck
        summary: DaemonSet rollout is stuck.
      expr: |-
        (
          (
            kube_daemonset_status_current_number_scheduled{namespace=~".*"}
              !=
            kube_daemonset_status_desired_number_scheduled{namespace=~".*"}
          ) or (
            kube_daemonset_status_number_misscheduled{namespace=~".*"}
              !=
            0
          ) or (
            kube_daemonset_updated_number_scheduled{namespace=~".*"}
              !=
            kube_daemonset_status_desired_number_scheduled{namespace=~".*"}
          ) or (
            kube_daemonset_status_number_available{namespace=~".*"}
              !=
            kube_daemonset_status_desired_number_scheduled{namespace=~".*"}
          )
        ) and (
          changes(kube_daemonset_updated_number_scheduled{namespace=~".*"}[5m])
            ==
          0
        )
      for: 15m
      labels:
        severity: warning
    - alert: KubeContainerWaiting
      annotations:
        description: Pod {{`{{ $labels.namespace }}`}}/{{`{{ $labels.pod }}`}} container {{`{{
$labels.container }}`}}
          has been in waiting state for longer than 1 hour.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubecontainerwaiting
        summary: Pod container waiting longer than 1 hour
      expr: sum by (namespace, pod, container, clusterName, environment)
(kube_pod_container_status_waiting_reason{namespace=~".*"}) > 0
      for: 1h
      labels:
        severity: warning
    - alert: KubeDaemonSetNotScheduled
      annotations:
        description: '{{`{{ $value }}`}} Pods of DaemonSet {{`{{ $labels.namespace }}`}}/{{`{{
$labels.daemonset }}`}} are not scheduled.'
```

```
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubedaemonsetnotscheduled
        summary: DaemonSet pods are not scheduled.
      expr: |-
        kube_daemonset_status_desired_number_scheduled{namespace=~".*"}
          -
        kube_daemonset_status_current_number_scheduled{namespace=~".*"} > 0
      for: 10m
      labels:
        severity: warning
    - alert: KubeDaemonSetMisScheduled
      annotations:
        description: '{{`{{ $value }}`}} Pods of DaemonSet {{`{{ $labels.namespace }}`}}/{{`{{
$labels.daemonset }}`}} are running where they are not supposed to run.'
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubedaemonsetmisscheduled
        summary: DaemonSet pods are misscheduled.
      expr: kube_daemonset_status_number_misscheduled{namespace=~".*"}
        > 0
      for: 15m
      labels:
        severity: warning
    - alert: KubeJobCompletion
      annotations:
        description: Job {{`{{ $labels.namespace }}`}}/{{`{{ $labels.job_name }}`}} is taking more
          than 12 hours to complete.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubejobcompletion
        summary: Job did not complete in time
      expr: kube_job_spec_completions{namespace=~".*"} - kube_job_status_succeeded{namespace=~".
*"}  > 0
      for: 12h
      labels:
        severity: warning
    - alert: KubeJobFailed
      annotations:
        description: Job {{`{{ $labels.namespace }}`}}/{{`{{ $labels.job_name }}`}} failed to
complete.
          Removing failed job after investigation should clear this alert.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubejobfailed
        summary: Job failed to complete.
      expr: kube_job_failed{namespace=~".*"}  > 0
      for: 15m
      labels:
        severity: warning
    - alert: KubeHpaReplicasMismatch
      annotations:
        description: HPA {{`{{ $labels.namespace }}`}}/{{`{{ $labels.horizontalpodautoscaler  }}`}}
          has not matched the desired number of replicas for longer than 15 minutes.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubehpareplicasmismatch
        summary: HPA has not matched descired number of replicas.
      expr: |-
        (kube_horizontalpodautoscaler_status_desired_replicas{namespace=~".*"}
          !=
        kube_horizontalpodautoscaler_status_current_replicas{namespace=~".*"})
          and
        (kube_horizontalpodautoscaler_status_current_replicas{namespace=~".*"}
          >
        kube_horizontalpodautoscaler_spec_min_replicas{namespace=~".*"})
          and
        (kube_horizontalpodautoscaler_status_current_replicas{namespace=~".*"}
          <
        kube_horizontalpodautoscaler_spec_max_replicas{namespace=~".*"})
          and
        changes(kube_horizontalpodautoscaler_status_current_replicas{namespace=~".*"}[15m]) == 0
      for: 15m
      labels:
        severity: warning
    - alert: KubeHpaMaxedOut
```

```
      annotations:
        description: HPA {{`{{ $labels.namespace }}`}}/{{`{{ $labels.horizontalpodautoscaler }}`}}
          has been running at max replicas for longer than 15 minutes.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubehpamaxedout
        summary: HPA is running at max replicas
      expr: |-
        kube_horizontalpodautoscaler_status_current_replicas{namespace=~".*"}
          ==
        kube_horizontalpodautoscaler_spec_max_replicas{namespace=~".*"}
      for: 15m
      labels:
        severity: warning
  - name: kubernetes-resources
    rules:
    - alert: KubeCPUOvercommit
      annotations:
        description: Cluster has overcommitted CPU resource requests for Pods and cannot
          tolerate node failure.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubecpuovercommit
        summary: Cluster has overcommitted CPU resource requests.
      expr: |-
        sum(namespace_cpu:kube_pod_container_resource_requests:sum{})
          /
        sum(kube_node_status_allocatable{resource="cpu"})
          >
        ((count(kube_node_status_allocatable{resource="cpu"}) > 1) - 1) / count
(kube_node_status_allocatable{resource="cpu"})
      for: 5m
      labels:
        severity: warning
    - alert: KubeMemoryOvercommit
      annotations:
        description: Cluster has overcommitted memory resource requests for Pods and
          cannot tolerate node failure.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubememoryovercommit
        summary: Cluster has overcommitted memory resource requests.
      expr: |-
        sum(namespace_memory:kube_pod_container_resource_requests:sum{})
          /
        sum(kube_node_status_allocatable{resource="memory"})
          >
        ((count(kube_node_status_allocatable{resource="memory"}) > 1) - 1)
          /
        count(kube_node_status_allocatable{resource="memory"})
      for: 5m
      labels:
        severity: warning
    - alert: KubeCPUQuotaOvercommit
      annotations:
        description: Cluster has overcommitted CPU resource requests for Namespaces.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubecpuquotaovercommit
        summary: Cluster has overcommitted CPU resource requests.
      expr: |-
        sum(kube_resourcequota{type="hard", resource="requests.cpu"})
          /
        sum(kube_node_status_allocatable{resource="cpu"})
          > 1.5
      for: 5m
      labels:
        severity: warning
    - alert: KubeMemoryQuotaOvercommit
      annotations:
        description: Cluster has overcommitted memory resource requests for Namespaces.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubememoryquotaovercommit
        summary: Cluster has overcommitted memory resource requests.
      expr: |-
```

```
        sum(kube_resourcequota{type="hard", resource="requests.memory"})
          /
        sum(kube_node_status_allocatable{resource="memory"})
          > 1.5
      for: 5m
      labels:
        severity: warning
    - alert: KubeQuotaAlmostFull
      annotations:
        description: Namespace {{`{{ $labels.namespace }}`}} is using {{`{{ $value }}`}} of its {{`
{{ $labels.resource }}`}} quota.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubequotaalmostfull
        summary: Namespace quota is going to be full.
      expr: |-
        kube_resourcequota{type="used"}
          / ignoring(instance, job, type)
        (kube_resourcequota{type="hard"} > 0)
          > 0.9 < 1
      for: 15m
      labels:
        severity: info
    - alert: KubeQuotaFullyUsed
      annotations:
        description: Namespace {{`{{ $labels.namespace }}`}} is using {{`{{ $value }}`}} of its {{`
{{ $labels.resource }}`}} quota.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubequotafullyused
        summary: Namespace quota is fully used.
      expr: |-
        kube_resourcequota{type="used"}
          / ignoring(instance, job, type)
        (kube_resourcequota{type="hard"} > 0)
          == 1
      for: 15m
      labels:
        severity: info
    - alert: KubeQuotaExceeded
      annotations:
        description: Namespace {{`{{ $labels.namespace }}`}} is using {{`{{ $value }}`}} of its {{`
{{ $labels.resource }}`}} quota.
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-kubequotaexceeded
        summary: Namespace quota has exceeded the limits.
      expr: |-
        kube_resourcequota{type="used"}
          / ignoring(instance, job, type)
        (kube_resourcequota{type="hard"} > 0)
          > 1
      for: 15m
      labels:
        severity: warning
    - alert: CPUThrottlingHigh
      annotations:
        description: '{{`{{ $value }}`}} throttling of CPU in namespace
          {{`{{ $labels.namespace }}`}} for container {{`{{ $labels.container }}`}} in pod {{`{{
$labels.pod }}`}}.'
        runbook_url: https://github.com/kubernetes-monitoring/kubernetes-mixin/tree/master/runbook.
md#alert-name-cputhrottlinghigh
        summary: Processes experience elevated CPU throttling.
      expr: |-
        sum(increase(container_cpu_cfs_throttled_periods_total{container!="", }[5m])) by
(environment, clusterName, container, pod, namespace)
          /
        sum(increase(container_cpu_cfs_periods_total{}[5m])) by (environment, clusterName,
container, pod, namespace)
          > ( 25 / 100 )
      for: 15m
      labels:
        severity: info
  - name: rds.rules
```

```
    rules:
    - alert: HighCpuUtilisation
      annotations:
        description: '{{`{{ printf "%.2f" $value }}`}} CPU utilisation has been found for RDS
instance: {{`{{ $labels.dbinstance_identifier }}`}}'
        summary: One or more RDS instances have high CPU utilisation
      expr: aws_rds_cpuutilization_average offset 12m > 90
      for: 15m
      labels:
        severity: warning
    - alert: LowFreeableMemory
      annotations:
        description: '{{`{{ printf "%.2f" $value }}`}} freeable memory has been found for RDS
instance: {{`{{ $labels.dbinstance_identifier }}`}}'
        summary: One or more RDS instance has low freeable memory
      expr: aws_rds_freeable_memory_average offset 12m < 1 *10^9
      for: 15m
      labels:
        severity: warning
    - alert: LowFreeStrorageSpace
      annotations:
        description: '{{`{{ printf "%.2f" $value }}`}} storage space has been found for RDS
instance: {{`{{ $labels.dbinstance_identifier }}`}}'
        summary: One or more RDS instance has low storage space
      expr: aws_rds_free_storage_space_average offset 12m < 10 * 10^9
      for: 15m
      labels:
        severity: warning
```

f. Restart the consul-template

```
service consul-template stop
service consul-template start
```

g. Restart Prometheus container

```
docker container stop promethues
docker container start promethues
```

h. Once the new rules are updated in SM Prometheus, proceed further.

3. Add below entries in consul Key/value.
   a. sensu/nkaas/interval
      i. value: 60
   b. sensu/nkaas/severity
      i. value: (critical|high|warning)
4. Update the sensu checks.
   a. SSH to the Monitoring Server.
   b. Modify the below files in order to pick the new Prometheus rules from the metric store.
   c. /etc/consul-template/sensu_container_checks.ctmpl

sensu_container_checks.ctmpl

```
<%-
check = scope.lookupvar('template_vars')
-%>
{
  {{ if key "sensu/platform" | regexMatch "^nkaas" }}
  {{- scratch.Set "init_runthrough" 0 -}}
  {{- $occurrences := key "sensu/nkaas/occurrences" -}}
  {{- $refresh := key "sensu/nkaas/refresh" -}}
  {{- $interval := key "sensu/nkaas/interval" -}}
  {{- $severity := key "sensu/nkaas/severity" -}}
  "checks": {
    {{- range $dc, $dckey := tree "sensu/datacenters/" | explode }}
        {{- if scratch.Get "init_runthrough" }},{{ end }}
        {{- $env := $dc }}
        "<%= check['category'] %>-{{ $env }}-alert": {
              "command": "/bin/ruby /opt/sensu-checks/query-prometheus.rb -h prometheus.service.
core.local -p 9090 -q \"ALERTS{environment=\\\"{{ $env }}\\\",severity=~\\\"{{ $severity }}\\\"}\"
-w 1 -c 1 -i true",
          "handlers": [
            "jira"
          ],
          "interval": {{ $interval }},
          "occurrences": {{ $occurrences }},
          "refresh": {{ $refresh }},
          "subscribers": [
            "roundrobin:<%= check['category'] %>"
          ],
          "standalone": true
  }
  {{- scratch.Set "init_runthrough" (add (scratch.Get "init_runthrough") 1) -}}
  {{- end -}} }
  {{- end -}}
}
```

/opt/jira_handler.rb

jira_handler.rb

```
#!/usr/bin/ruby
## Capgemini UK PLC Proprietary and Confidential ##
## Copyright Capgemini 2019 - All Rights Reserved ##

############################################################
# Title:        Jira Handler Script
# Description:  This script will live on the sensu server and create automated notifications for
our service desk
# API calls:
# Retrieve Service Desk ID => GET /rest/servicedeskapi/servicedesk
# Retrieve Request Type ID => GET /rest/servicedeskapi/servicedesk/{serviceDeskId}/requesttype
# Create Automed Ticket    => POST /rest/servicedeskapi/request
# {
#   "serviceDeskId": "10",
#   "requestTypeId": "25",
#   "requestFieldValues": {
#     "summary": "There is an issue on this server with this service."
#   }
# }
# Retrieve Existing Ticket => GET /rest/servicedeskapi/request/{issueIdOrKey}
############################################################

require 'rubygems'
require 'json'
require 'timeout'
require 'time'
```

```
require 'logger'

def setupLogging()
  $logger = Logger.new('/var/log/jira_handler/jira_handler.log', 10, 1024000)

  case $loggerLevel
  when "debug"
    $logger.level = Logger::DEBUG
  when "info"
    $logger.level = Logger::INFO
  end
end

def readSensuInput(file = STDIN)
  @event = ::JSON.parse(file.read)
  @event['occurrences'] ||= 1
  @event['check']       ||= Hash.new
  @event['client']      ||= Hash.new
rescue => e
  puts 'error reading event: ' + e.message
  exit 0
end

def getCheckName()
  @checkName ||= @event['check']['name'].to_s
  $logger.debug("Check name #{@checkName}")
  return @checkName
end

# Adding delays base on the node count to avoid all monitoring nodes running at the same time
def addNodeDelay()
  hostname = `hostname`
  nodeCount = hostname.strip.split("-").last

  if nodeCount !~ /initial/
    nodeCount = 1+nodeCount.to_i
    delay = nodeCount*60
    $logger.info("Delayed by #{delay} seconds" )
    sleep(delay)
  end
end

def getConsulKeyPath(ticketKey)
  return "serviceDesk/tickets/#{ticketKey}-TicketId"
end

def getExistingTicketId(ticketKey)
  consulKeyPath = getConsulKeyPath(ticketKey)
  $logger.debug("Checking for existing ticket ID under consul KV path: #{consulKeyPath}")
  $jiraTicketId = `consul kv get #{consulKeyPath}`.strip
  if $jiraTicketId == ""
    @newTicket = true
    $logger.debug("Existing ticket not found under consul KV path: #{consulKeyPath}")
  else
    $logger.debug("Found existing ticket ID: #{$jiraTicketId}")
  end
end

# Ensures that the Jira Service Desk API is available.
def isJiraResponsive()
  "200" == Timeout::timeout(50) do
     `curl -X GET -s -o /dev/null -w "%{http_code}" -H "Content-Type:application/json" -H
"Authorization:Basic #{$authKey}" "#{$jiraAddress}"/rest/servicedeskapi/info`
   end
end

# Attempts to determine whether there is a ticket created already
def shouldTicketBeCreated()
  if @newTicket
    $logger.info("Creating new ticket for alert as an existing one does not exist")
    return true
```

```
  end

  ticketSearch = Timeout::timeout(50) do
    `curl -s -X GET -H "Content-Type:application/json" -H "Authorization:Basic #{$authKey}" "#
{$jiraAddress}"/rest/servicedeskapi/request/"#{$jiraTicketId}"`
  end

  $logger.debug("Ticket search response: #{ticketSearch}")
  errorMatch = ticketSearch.match(/<title>(.*?)<\/title>/)

  unless errorMatch.nil?
    $logger.debug("Error match: #{errorMatch}")
    $logger.error("Error checking for existing ticket #{$jiraTicketId}, new ticket will not be
created: #{errorMatch[1]}")
    return false
  end

  if ticketSearch.include?('Cannot find issue')
    $logger.info("Creating new ticket for alert as the existing one #{$jiraTicketId} cannot be
found in Jira Service Desk")
    return true
  end

  ticketStatus = JSON.parse(ticketSearch)['currentStatus']['status']

  $logger.debug("Existing ticket status is " + ticketStatus.to_s)
  if ticketStatus == "Open" || ticketStatus == "In Progress" || ticketStatus == "Work in progress"
|| ticketStatus == "Pending"
    $logger.info("There is an open ticket already created #{$jiraTicketId}, a new one will not be
created")
    return false
  else
    $logger.info("The current ticket #{$jiraTicketId} is not open a new ticket will be created")
    return true
  end
end

def makeJiraIssueJson(checkName = getCheckName(), severity = $outputSeverity, detail =
$outputDetails)
  jiraDataHash["serviceDeskId"] = $serviceDeskId.to_s
  jiraDataHash["requestTypeId"] = $requestTypeId.to_s
  jiraDataHash["requestFieldValues"]["summary"] = "Automated Ticket #{consulDataCentre} - #
{checkName}"

  description = "||SM Environment|#{consulDataCentre}||\n"
  description << "||Check Name|#{checkName}||\n"
  description << "||Check Severity|#{severity}||\n"

  description << "h4. Output:\n"

  if severity.start_with?("PrometheusData")
     prometheusOutputDetails = eval(detail)
     prometheusOutputDetails.each do |details|
       sortedDetails = details.sort_by { |hsh| hsh['Name'] }
       sortedDetails.each do |detail|
         description << "|#{detail['Name']}|#{detail['Value']}|\n"
       end
       description << "----\n"
     end
  else
    description << "{code:json}#{detail}{code}\n"
  end

  description << "\n"

  description << "h4. Command:\n{code:bash}#{@event['check']['command']}{code}\n"
  description << "h4. History:\n{code:bash}#{@event['check']['history']}{code}\n"
  $logger.debug("JIRA ticket description: #{description}")
  jiraDataHash["requestFieldValues"]["description"] = description

  jiraData=JSON.generate(jiraDataHash)
```

```
  $logger.debug("JSON data for jiraDataHash is " + jiraData.to_s)
  jiraData
end

def consulDataCentre
  JSON.parse(consulCatalogNode)['Config']['Datacenter']
end

def consulCatalogNode
  `curl -s http://#{$consulAddress}/v1/agent/self`
end

def jiraDataHash
  @jiraDataHash ||= Hash.new { |h,k| h[k] = Hash.new(&h.default_proc) }
end

# Attempts to create the Jira Ticket
def createTicket (jiraData, consulKeyPath, saveTicketId=true)
  ticket = Timeout::timeout(50) do
    `curl -s -X POST -H "Content-Type:application/json" -H "Authorization:Basic #{$authKey}" --
data '#{jiraData}' "#{$jiraAddress}"/rest/servicedeskapi/request`
  end

  $logger.debug("Create ticket response: #{ticket}")
  ticketId = JSON.parse(ticket)['issueKey']
  $logger.debug("Ticket has been created successfully - #{ticketId}")

  if saveTicketId
    `consul kv put #{consulKeyPath} #{ticketId}` # Key of the already created issue
  end
end

def handlePrometheusAlerts()
  ticketKeyPrefix = getCheckName()
  prometheusOutputDetails = eval($outputDetails)

  # loop over each of the different alerts in the prometheus data
  prometheusOutputDetails.each do |details|
    $logger.debug("Processing: #{details}")

    ticketAlertname = ""
    ticketNamespace = ""
    ticketSuffix = ""
    outputDetail = "{\n"

    # loop over each of the labels for the current alert and build up the ticket key and
description
    # the labels are sorted to ensure the ordering of the ticket key parts to avoid duplicates
    sortedDetails = details.sort_by{ |kv| kv['Name'] }
    sortedDetails.each do |labelSet|
      labelName = labelSet['Name'].to_s
      labelValue = labelSet['Value'].to_s
      $logger.debug("Name: #{labelName}, Value: #{labelValue}")

      outputDetail << "  \"#{labelName}\": \"#{labelValue}\",\n"

      # skip the labels that aren't significant or we have already added
      # the alertname and namespace are included without the label name
      if labelName == "alertname"
        ticketAlertname = "-#{labelValue.to_s.downcase}"
      elsif labelName == "namespace"
        ticketNamespace = "-#{labelValue.to_s.downcase}"
      elsif labelName !~ /
(alertname|namespace|instance|severity|clusterName|environment|job|__name__|alertstate)/
        ticketSuffix  << '-' << labelName.to_s.downcase << '-' << labelValue.to_s.downcase
      end
    end

    outputDetail = outputDetail.chomp(",\n")
    outputDetail << "\n}"
```

```
    ticketKey = ticketKeyPrefix + ticketAlertname + ticketNamespace + ticketSuffix

    $logger.debug("ticketKeyPrefix: #{ticketKeyPrefix}")
    $logger.debug("ticketAlertname: #{ticketAlertname}")
    $logger.debug("ticketNamespace: #{ticketNamespace}")
    $logger.debug("ticketSuffix: #{ticketSuffix}")

    $logger.debug("ticketKey: #{ticketKey}")
    $logger.debug("outputDetail: #{outputDetail}")

    getExistingTicketId(ticketKey)

    if shouldTicketBeCreated()
      createTicket(makeJiraIssueJson(ticketKey, $outputSeverity.sub(/^PrometheusData/, ''),
outputDetail), getConsulKeyPath(ticketKey))
    end
  end
end

if __FILE__ == $0
  # Passed in arguments
  $jiraAddress = ARGV[0] # URL of Jira Service Desk e.g. https://enterpriseipaasdemo1.atlassian.
net/
  $loggerLevel = ARGV[1] # info or debug
  $consulAddress = ARGV[2] # the hostname:port of the consul api e.g. 'localhost:8500'

  setupLogging()
  addNodeDelay()
  readSensuInput()
  $logger.debug("The failing check is: #{@event['check']['name']}")
  $logger.debug("Full event: #{@event}")

  if @event.has_key?('check')
    status = @event['check']['status'].to_i
    if (status == 3) || (status == 0)
      $logger.info("Event status #{status}, handler will not execute. Sensu Output: #{@event
['check']['output']}")
      $logger.close
      exit 2
    end
  else
    $logger.error("event.check.status key does not exist, event invalid: #{@event}")
    exit 3
  end

  $logger.debug("Getting values from consul")

  # Numeric ID that coordinates with the particular service desk we wish to raise a ticket in
  $serviceDeskId = `consul kv get serviceDesk/serviceDeskId`.strip
  $logger.debug("serviceDeskId: #{$serviceDeskId}")

  # Numeric ID that corresponds to the type of ticket we want to raise
  $requestTypeId = `consul kv get serviceDesk/requestTypeId`.strip
  $logger.debug("requestTypeId: #{$requestTypeId}")

  # base 64 encoded username and password
  $authKey = `consul kv get serviceDesk/authKey`.strip
  $logger.debug("authKey: #{$authKey}" )

  # monitoring platform type
  $platform = `consul kv get sensu/platform`.strip
  $logger.debug("platform: #{$platform}" )

  unless isJiraResponsive()
    $logger.info("Jira Unresponsive or erroring, exiting script")
    exit 2
  else
    $logger.info("Jira responding correctly, continuing with script")
  end

  $output = "#{@event['check']['output']}"
```

```
  $outputSeverity = $output.match(": ").pre_match
  $outputDetails = $output.match(": ").post_match
  $logger.debug("Output is " + $outputSeverity + "\n" + $outputDetails)

  # For v2 (EKS) platforms there is a single prometheus query for the alerts that needs to be
  # processed into individual possible Jira tickets
  if $outputSeverity.start_with?("PrometheusData") && $platform =~ /^nkaas/
    $logger.info("Handling prometheus alert ticket creation flow")
    handlePrometheusAlerts()
  else
    # Follow the flow for a single ticket based on the alert
    $logger.info("Handling single ticket creation flow")
    ticketKey = getCheckName()
    getExistingTicketId(ticketKey)
    if shouldTicketBeCreated()
      createTicket(makeJiraIssueJson(), getConsulKeyPath(ticketKey))
    end
  end

  $logger.close
end
```

/etc/consul.d/seed_kv/sensu/nkaas.json

nkaas.json

```
[
  {
    "key": "sensu/nkaas/occurrences",
    "flag": 0,
    "value": "3"
  },
  {
    "key": "sensu/nkaas/refresh",
    "flag": 0,
    "value": "1800"
  },
  {
    "key": "sensu/nkaas/interval",
    "flag": 0,
    "value": "60"
  },
  {
    "key": "sensu/nkaas/severity",
    "flag": 0,
    "value": "(critical|high|warning)"
  },
  {
    "key": "sensu/nkaas/xpaas-ns/pod-cpu-critical",
    "flag": 0,
    "value": "80"
  },
  {
    "key": "sensu/nkaas/xpaas-ns/pod-cpu-warning",
    "flag": 0,
    "value": "70"
  },
  {
    "key": "sensu/nkaas/xpaas-ns/pod-cpu-interval",
    "flag": 0,
    "value": "120"
  },
  {
    "key": "sensu/nkaas/xpaas-ns/pod-failures-critical",
    "flag": 0,
    "value": "1"
  },
  {
```

```
  "key": "sensu/nkaas/xpaas-ns/pod-failures-warning",
  "flag": 0,
  "value": "1"
},
{
  "key": "sensu/nkaas/xpaas-ns/pod-failures-interval",
  "flag": 0,
  "value": "120"
},
{
  "key": "sensu/nkaas/xpaas-ns/container-waiting-critical",
  "flag": 0,
  "value": "1"
},
{
  "key": "sensu/nkaas/xpaas-ns/container-waiting-warning",
  "flag": 0,
  "value": "1"
},
{
  "key": "sensu/nkaas/xpaas-ns/container-waiting-interval",
  "flag": 0,
  "value": "120"
},
{
  "key": "sensu/nkaas/cp/pod-mem-critical",
  "flag": 0,
  "value": "90"
},
{
  "key": "sensu/nkaas/cp/pod-mem-warning",
  "flag": 0,
  "value": "80"
},
{
  "key": "sensu/nkaas/cp/pod-mem-interval",
  "flag": 0,
  "value": "120"
},
{
  "key": "sensu/nkaas/client-ns/pod-cpu-critical",
  "flag": 0,
  "value": "80"
},
{
  "key": "sensu/nkaas/client-ns/pod-cpu-warning",
  "flag": 0,
  "value": "70"
},
{
  "key": "sensu/nkaas/client-ns/pod-cpu-interval",
  "flag": 0,
  "value": "120"
},
{
  "key": "sensu/nkaas/client-ns/pod-failures-critical",
  "flag": 0,
  "value": "1"
},
{
  "key": "sensu/nkaas/client-ns/pod-failures-warning",
  "flag": 0,
  "value": "1"
},
{
  "key": "sensu/nkaas/client-ns/pod-failures-interval",
  "flag": 0,
  "value": "120"
},
{
  "key": "sensu/nkaas/client-ns/container-waiting-critical",
```

```
    "flag": 0,
    "value": "1"
  },
  {
    "key": "sensu/nkaas/client-ns/container-waiting-warning",
    "flag": 0,
    "value": "1"
  },
  {
    "key": "sensu/nkaas/client-ns/container-waiting-interval",
    "flag": 0,
    "value": "120"
  },
  {
    "key": "sensu/nkaas/worker/pod-mem-critical",
    "flag": 0,
    "value": "90"
  },
  {
    "key": "sensu/nkaas/worker/pod-mem-warning",
    "flag": 0,
    "value": "80"
  },
  {
    "key": "sensu/nkaas/worker/pod-mem-interval",
    "flag": 0,
    "value": "120"
  },
  {
    "key": "prometheus/services/master.kubelet",
    "flag": 0,
    "value": ""
  },
  {
    "key": "prometheus/services/worker.kubelet",
    "flag": 0,
    "value": ""
  },
  {
    "key": "sensu/platform",
    "flag": 0,
    "value": "nkaas"
  }
]
```

d. Once the above files are updated then restart the consul-template.

```
service consul-template stop
service consul-template start
```

e. Once you restart consul-template, sensu-server/api/client will automatically get restarted.

5. Add prometheus-cloudwatch-exporter addon to k8's cluster.
   a. By using this addon we can export the cloud watch logs from AWS to cluster.
   b. Connect to the Ops VPN and access Rundeck.
   c. Go to Rundeck, Run Provision > Kubernetes > Deploy Kubernetes Addon with the below values.
      i. Environment_region eu-west-1
      ii. Environment management
      iii. Addon prometheus-cloudwatch-exporter
      iv. Action All
   d. Run post actions for the 'xpaas' addon by using the 'provision/kubernetes/Deploy Kubernetes Addon' job in Rundeck.
   e. Edit the cluster role to include the 'prometheus-cloudwatch-exporter:9106' resource
      i. Edit the cluster role.

```
k edit clusterrole prometheus
```

      ii. Add the prometheus-cloudwatch-exporter:9106" resource under resourceNames section same as below.

```
resourceNames:
- kube-state-metrics:8080
- prometheus-cloudwatch-exporter:9106
```

f. Once the addon has been deployed you will need to also run the redeploy_prometheus rundeck job so that it will pick up the new target.
g. Go to Rundeck, Run Platform Maintainance Redeploy Prometheus with the below values.
    i. imageLabel v2.10.0-xpaas-1.0.0
    ii. memory_limit 2g
    iii. cpu_limit 1.5

> The above values you'll get from the "docker inspect prometheus" command.
>
> For imageLabel: "docker inspect prometheus | grep -i version"
>
> For cpu_limit: "docker inspect prometheus | grep -i cpu"
>
> 1000000000 NanoCPU = 1CPU
>
> For memory_limit = "docker inspect prometheus | grep -i mem"

6. Update RDS Tag.
    a. Login to the DEMO AWS RDS console.
    b. You will need to tag any RDS instances that you want to monitor with 'Monitor_Metrics: ["true"]'
    c. Currently, this tag is set to False.
    d. Tag you can find under
        i. Databases > <environment>-alm-database > Tag.
    e. Create an IAM policy with the below details.
        i. Name: <environment>-cloudwatch_exporter-policy
        ii. services: CloudWatch(access level= List and Read), and Resource Group Tagging(access level= Read).
        iii. Then attach the <environment>-kube_worker role to the above policy.
    f. Then allow 9106 port on <client-prefix>-kubernetes-worker-management security group.
        i. Add new rule
        ii. TCP with 9106 port number.
        iii. Source: <client-prefix>-kubernetes-cp-management SG ID.
    g. Tag you can find under

Post-implementation Checks

1. Check the SM Prometheus status.
2. Make sure newly created rules are present on SM Prometheus.
3. Check the status of the sensu.
4. Check if k8's related alerts are triggering on service desk and make sure all alerts are individual w.r.t. Prometheus alerts.
5. Check the cluster status.
6. Check the consul services.
7. Make sure Prometheus/kibana/Grafana are working fine.

Backout plan

1. Update the Prometheus rules
    a. SSH to the Metric store.
    b. Update the below file to normal.

```
cp /etc/consul-template/prometheus_rules.ctmpl-bkp /etc/consul-template/prometheus_rules.ctmpl-bkp
cp /etc/consul-template/prometheus.ctmpl-bkp /etc/consul-template/prometheus.ctmpl
```

    c. Restart the consul-template.

```
service consul-template stop
service consul-template start
```

2. Update the sensu checks.
    a. SSH to the Monitoring server.
    b. Modify the below files.

```
cp /etc/consul-template/sensu_container_checks.ctmpl-bkp /etc/consul-template
/sensu_container_checks.ctmpl
cp /opt/jira_handler.rb-bkp /opt/jira_handler.rb
cp /etc/consul.d/seed_kv/sensu/nkaas.json-bkp /etc/consul.d/seed_kv/sensu/nkaas.json
```

c. Once the above files are updated then restart the consul-template.

```
service consul-template stop
service consul-template start
```

d. Once you restart consul-template, sensu-server/api/client will automatically get restarted.

3. Remove prometheus-cloudwatch-exporter addon from k8's cluster.
    a. Make the Prometheus-cloudwatch-exporter replicas to 0 in deployment.
4. Update RDS Tag.
    a. Login to DEMO AWS RDS console.
    b. Change the tag with 'Monitor_Metrics: ["false"]'
    c. Tag you can find under
    d. Databases > demo-management-alm-database > Tags
    e. Tag you can find under
        i. Databases > <environment>-alm-database > Tag.
    f. Remove below IAM policy.
        i. Name: <environment>-cloudwatch_exporter-policy.
    g. Then remove 9106 port on <client-prefix>-kubernetes-worker-management security group.
        i. TCP with 9106 port number.
        ii. Source: <client-prefix>-kubernetes-cp-management SG ID.